An Optimal Local Approximation Algorithm for Max-Min Linear Programs

Patrik Floréen
patrik.floreen@cs.helsinki.fi

Joel Kaasinen
joel.kaasinen@cs.helsinki.fi

Petteri Kaski
petteri.kaski@cs.helsinki.fi

Jukka Suomela
jukka.suomela@cs.helsinki.fi

Helsinki Institute for Information Technology HIIT, University of Helsinki
P.O. Box 68, FI-00014 University of Helsinki

ABSTRACT

In a max-min LP, the objective is to maximise ω subject to $A\mathbf{x} \le \mathbf{1}$, $C\mathbf{x} \ge \omega\mathbf{1}$, and $\mathbf{x} \ge \mathbf{0}$ for nonnegative matrices A and C. We present a local algorithm (constant-time distributed algorithm) for approximating max-min LPs. The approximation ratio of our algorithm is the best possible for any local algorithm; there is a matching unconditional lower bound.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; G.1.6 [**Numerical Analysis**]: Optimization —*linear programming*

General Terms

Algorithms, Theory

1. INTRODUCTION

In a *max-min linear program* (max-min LP), the objective is to

$$\begin{aligned} \text{maximise} \quad & \min_{k \in K} \mathbf{c}_k \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \le \mathbf{1}, \\ & \mathbf{x} \ge \mathbf{0} \end{aligned} \tag{1}$$

or, equivalently, to

$$\begin{aligned} \text{maximise} \quad & \omega \\ \text{subject to} \quad & A\mathbf{x} \le \mathbf{1}, \\ & C\mathbf{x} \ge \omega\mathbf{1}, \\ & \mathbf{x} \ge \mathbf{0}. \end{aligned}$$

The matrices A and C are nonnegative and sparse: each row $\mathbf{a}_i$ of A has at most Δ_I positive elements, and each row $\mathbf{c}_k$ of C has at most Δ_K positive elements for some constants Δ_I and Δ_K.

This work studies the task of (approximately) solving a max-min LP that is distributed to a network of nodes, with one node $v \in V$ for each variable x_v, one node $i \in I$ for each constraint $\mathbf{a}_i\mathbf{x} \le 1$, and one node $k \in K$ for each objective $\mathbf{c}_k\mathbf{x}$. Nodes $v \in V$ and $i \in I$ are adjacent if a_{iv} is positive, and nodes $v \in V$ and $k \in K$ are adjacent if c_{kv} is positive. Each node $v \in V$ is to choose the value x_v by exchanging messages with its adjacent nodes.

Immediate applications of this setup include various tasks of fair resource allocation in contemporary networking, such as fair bandwidth allocation in a communication network and balanced data gathering in a wireless sensor network. For example, a solution to a max-min LP may represent feasible data flows $\mathbf{x}$ that maximise the minimum amount of bandwidth ω that we provide to each customer $k \in K$. An algorithm for approximating max-min LPs also enables one to solve approximate mixed packing and covering LPs [20]; a particular special case is finding an (approximate) solution to a nonnegative system of linear equations.

The main contribution of this paper is a distributed algorithm that achieves the best possible approximation ratio for max-min LPs in the context of *local algorithms* [13, 15, 18], that is, distributed algorithms that complete in *constant time* (constant number of synchronous communication rounds), independent of the size of the network. The corresponding nonexistence result, which we repeat in our main theorem below, appears in earlier work [7]. The cases $\Delta_I = 1$ or $\Delta_K = 1$ can be solved optimally with a local algorithm [17]; we focus on non-trivial max-min LPs with $\Delta_I \ge 2$ and $\Delta_K \ge 2$.

THEOREM 1. *For any $\Delta_I \ge 2$, $\Delta_K \ge 2$, and $\epsilon > 0$, there is a local approximation algorithm for the max-min LP problem with the approximation ratio $\Delta_I(1 - 1/\Delta_K) + \epsilon$. Moreover, there is no local approximation algorithm for max-min LPs with the approximation ratio $\Delta_I(1 - 1/\Delta_K)$.*

To our knowledge, this is the first example of a "nontrivial" approximation threshold in the context of local algorithms; in particular, it is not obvious that a threshold should exist at precisely $\Delta_I(1 - 1/\Delta_K)$. Moreover, the threshold is combinatorial in the sense that it is independent of the coefficients in the constraints and the objectives. In fact, the inapproximability result holds for max-min LPs with $\{0, 1\}$ coefficients [7].

1.1 Definitions

We give a formal definition of a max-min LP in a distributed setting. Let $\mathcal{G} = (V \cup I \cup K, E)$ be a bipartite, undirected communication graph. The nodes $v \in V$ are called *agents*, the nodes $i \in I$ are called *constraints*, and the nodes $k \in K$ are called *objectives*; the sets V, I, and K are pairwise disjoint. Each edge $e \in E$ is of the form $e = \{v, i\}$ or $e = \{v, k\}$ where $v \in V$, $i \in I$, and $k \in K$.

Let

$$\begin{aligned} V_i &= \{v \in V : \{v, i\} \in E\}, \\ V_k &= \{v \in V : \{v, k\} \in E\}, \\ I_v &= \{i \in I : \{v, i\} \in E\}, \\ K_v &= \{k \in K : \{v, k\} \in E\} \end{aligned}$$

for all $i \in I$, $k \in K$, and $v \in V$. We assume that $|V_i| \le \Delta_I$ and $|V_k| \le \Delta_K$ for all $i \in I$ and $k \in K$ for some constants Δ_I and Δ_K.

A *max-min linear program* associated with $\mathcal{G}$ is defined as follows. Associate a variable x_v with each agent $v \in V$, associate a coefficient $a_{iv} > 0$ with each edge $\{i, v\} \in E$, $i \in I$, $v \in V$, and associate a coefficient $c_{kv} > 0$ with each edge $\{k, v\} \in E$, $k \in K$, $v \in V$. The task is to

$$\begin{aligned} \text{maximise} \quad & \omega(\mathbf{x}) = \min_{k \in K} \sum_{v \in V_k} c_{kv} x_v \\ \text{subject to} \quad & \sum_{v \in V_i} a_{iv} x_v \le 1, \quad \forall\, i \in I, \\ & x_v \ge 0, \quad \forall\, v \in V. \end{aligned} \tag{2}$$

The *local input* of an agent $v \in V$ consists of the sets I_v and K_v and the coefficients a_{iv}, c_{kv} for all $i \in I_v, k \in K_v$. The local input of a constraint $i \in I$ consists of V_i, and the local input of an objective $k \in K$ consists of V_k.

1.2 Model of distributed computation

Each node in the communication graph $\mathcal{G}$ is a computational entity. During each synchronous communication round, each node in parallel (i) performs local computation, (ii) sends a message to each neighbour, and (iii) receives a message from each neighbour. Eventually, after D communication rounds, each agent $v \in V$ in parallel produces the output x_v, and the algorithm stops.

In a local algorithm, we assume that D is a constant. The value of D may depend on the parameters Δ_I and Δ_K and the desired approximation ratio, but it is independent of the number of nodes in the network. The constant D is called the *local horizon* of the algorithm. For each agent $v \in V$, the output x_v is a function of the local inputs of the nodes within distance D (in number of edges) from v in the communication graph $\mathcal{G}$.

The inapproximability part of Theorem 1 holds even if we assume that each node of the graph $\mathcal{G}$ has a unique identifier, while our approximation algorithm does not need any node identifiers. We merely assume *port numbering* [1, 3, 19]: each node chooses an ordering on its incident edges.

1.3 Prior work

Local algorithms are scalable and fault-tolerant [14, 15] and hence highly desirable from a practical networking perspective. A local algorithm completes in constant time in an arbitrarily large network, and changes in the input at one node only affects the output in its radius-D neighbourhood. Indeed, in bounded-degree graphs, a local algorithm is also a dynamic graph algorithm (with constant-time updates) and a self-stabilising algorithm (with constant time to stabilise). For more information on the model of local algorithms and their advantages, we refer to the survey [18].

For any $\epsilon > 0$, there exists a local $(1+\epsilon)$-approximation algorithm for packing and covering LPs, assuming a bounded-degree graph and bounded coefficients [10, 11]. However, as shown in Theorem 1, this is not the case with max-min LPs [7].

Two special cases of max-min LPs have been considered in prior work: bipartite max-min LPs and max-min LPs with $\{0, 1\}$ coefficients. (In a *bipartite* max-min LP, each column of A and each column of C contains only one nonzero element, that is, each agent $v \in V$ is adjacent to exactly one constraint $i \in I$ and exactly one objective $k \in K$.)

The inapproximability part of Theorem 1 holds in the case of bipartite max-min LPs with $\{0, 1\}$ coefficients [7]. It is known from prior work [6, 7] that a local algorithm can achieve the approximation factor $\Delta_I(1 - 1/\Delta_K) + \epsilon$ for any $\epsilon > 0$ in bipartite max-min LPs, and in max-min LPs with $\{0, 1\}$ coefficients and $\Delta_K = 2$. However, these techniques apparently do not extend; for general max-min LPs, the best local algorithm in prior work is the *safe algorithm* which achieves the factor Δ_I approximation [8, 16]. The present work extends prior work by showing that the approximation factor $\Delta_I(1 - 1/\Delta_K) + \epsilon$ can be achieved for arbitrary max-min LPs.

Max-min LPs provide, to our knowledge, the first example of a natural problem where there are matching, nontrivial lower and upper bounds for the approximation factor of a deterministic local algorithm. Recently, another pair of matching lower and upper bounds for local algorithms has been discovered: in bounded-degree graphs, there is a local 2-approximation algorithm for the vertex cover problem [2], and there is no local algorithm with the approximation ratio $2 - \epsilon$ for any $\epsilon > 0$ [5, 12].

2. OVERVIEW OF THE ALGORITHM

We begin in §3 by reminding that, in the port numbering model, we can without loss of generality focus on the case where the graph $\mathcal{G}$ is a (countably infinite but locally finite) tree [1, 7].

In §4, we present a series of local transformations that simplify the structure of the problem. We show that with the objective of establishing Theorem 1, it is sufficient to focus on the special case where $|V_i| = 2$, $|V_k| \ge 2$, $|K_v| = 1$, $|I_v| \ge 1$, and $c_{ku} = 1$ for all $i \in I$, $k \in K$, $v \in V$, $u \in V_k$. Figure 1 shows an example of a communication graph $\mathcal{G}$ after the local transformations.

In §5, we present a local algorithm for this special case. In §6, we prove that the output of the algorithm is a factor $2(1 - 1/\Delta_K) + \epsilon'$ approximation. Put together, we have a local, factor $\Delta_I(1 - 1/\Delta_K) + \epsilon$ approximation for general max-min LPs, for any $\epsilon > 0$.

The intuition behind the algorithm in §5 is best understood if we study its analysis in §6. In the analysis, it is convenient to assume that we have assigned a one-dimensional coordinate, *layer*, to each node of the tree $\mathcal{G}$; see Figure 1 for an example. When we assign the layers, we also partition the agents into *up-agents* and *down-agents*. We have alternatingly layers of up-agents, constraints, down-agents, and

node type	layer	level in $\mathcal{A}_u$	solution of $\mathcal{A}_u$
passive down-agents	-3		
constraints	-2	-2	
up-agents	-1	-1	$f^-_{u,\cdot,1}$
objectives	0	0	
down-agents	1	1	$f^+_{u,\cdot,1}$
constraints	2	2	
up-agents	3	3	$f^-_{u,\cdot,0}$
objectives	4	4	
down-agents	5	5	$f^+_{u,\cdot,0}$
constraints	6	6	
passive up-agents	7		
passive objectives	8		
passive down-agents	9		
constraints	10		
up-agents	11		

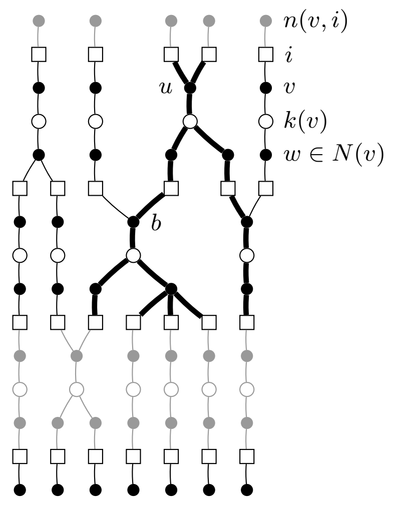

Figure 1: The graph $\mathcal{G}$ (after the local transformations in §4) and the layers (see §6). We have chosen $R = 3$ and hence $r = 1$. Thick lines highlight the tree $\mathcal{A}_u$ (see §5.1). There are several ways to choose the layers; nevertheless, if u is an up-agent on layer -1 then the levels in $\mathcal{A}_u$ necessarily coincide with the layers.

objectives. Each objective has exactly one adjacent up-agent "above" it, and at least one adjacent down-agent "below" it; hence the names "up" and "down".

We now assign every R^{th} layer of objectives to be *passive*, including the adjacent agents that set $x_v = 0$. Each agent v computes an upper bound t_v of the optimum; see §5.2. Then we construct a solution for the active layers in a greedy manner, starting with a layer of passive up-agents and propagating information upwards until we reach the next layer of passive down-agents; see §5.3.

This is not yet an approximation for the original problem: while most objectives perform at least as well as in the global optimum, the passive objectives have utility 0. By applying ideas from the shifting strategy [4, 9], we could consider R possible choices for the locations of the passive layers. Then we could take averages over these to obtain a solution $\mathbf{y}$. In §6.2 we show that $\mathbf{y}$ would indeed be a factor $R/(R-1)$ approximation.

There is one difficulty, however. We cannot assign the layers by a local algorithm in a globally consistent manner; in particular, we do not know whether a given agent is a down-agent or an up-agent. To overcome this, we consider both possible roles for each agent, up and down. For both roles, we compute a candidate solution by applying the shifting strategy. Finally we take the average of both candidate solutions. This is the essence of (18). In §6, we prove that this local approach yields a globally feasible solution that is within factor $2(1 - 1/\Delta_K) + \epsilon'$ of the optimum. The constant $\epsilon' > 0$ can be made arbitrarily small by choosing a sufficiently large R.

Yet another basic hurdle implicitly overcome in the proof of Theorem 1 stems from *underconstrained instances*: if there are several equally good solutions, one needs to choose between them in a globally consistent manner. Our definition of the values g^+ and g^- in (12)–(14) addresses this by focusing on a particular extreme point. Each layer of down-agents chooses as large values g^+ as possible, without violating the constraints "below" them. Each layer of up-agents chooses as small values g^- as possible, as long as the objectives "below" them meet the smoothed upper bounds s_v.

3. PORT NUMBERING AND UNFOLDING

Our algorithm does not need to use any node identifiers; port numbering is sufficient. In the port numbering model, a local algorithm cannot distinguish between a short cycle and an infinitely long path. We can exploit this limitation to simplify the description of our local algorithm: we can assume that we have *unfolded* all cycles of the graph $\mathcal{G}$ [1, 7], as follows.

A *walk* of *length* ℓ in a graph $\mathcal{G}$ is a nonempty tuple $(u_0, e_1, u_1, e_2, u_2, \dots, e_\ell, u_\ell)$ of alternating nodes and edges in $\mathcal{G}$ such that, for all $j = 1, 2, \dots, \ell$, the edge e_j joins the nodes u_{j-1} and u_j. The walk is said to *start* at u_0 and *end* at u_ℓ. A walk is *non-backtracking* if $e_{j-1} \neq e_j$ holds for all $j = 1, 2, \dots, \ell$. A *path* is a walk with no repeated nodes.

Let $\mathcal{G}$ be a finite connected graph and let r be a node of $\mathcal{G}$. The *unfolding* of $\mathcal{G}$ *rooted at* r is the undirected simple graph $\mathcal{G}'$ obtained as follows. The node set of $\mathcal{G}'$ is the set of all non-backtracking walks in $\mathcal{G}$ that start at r. Two nodes of $\mathcal{G}'$ are joined by an edge iff one can be obtained from the other by appending exactly one edge and one node of $\mathcal{G}$.

We associate with each node of $\mathcal{G}'$ a *parent* node of $\mathcal{G}$, namely the end-node of the walk. We also associate with each edge of $\mathcal{G}'$ a *parent* edge of $\mathcal{G}$, namely the appended edge.

Remarks.

1. The unfolding $\mathcal{G}'$ is a tree.
2. The unfolding $\mathcal{G}'$ is finite iff $\mathcal{G}$ is a tree; otherwise $\mathcal{G}'$ is countably infinite.
3. Any two unfoldings of $\mathcal{G}$ rooted at different nodes are isomorphic. In what follows we refer to "the" unfolding of $\mathcal{G}$ without specifying a particular root node.

4. Assuming that the graph $\mathcal{G}$ has port numbers associated with the ends of its edges, the unfolding $\mathcal{G}'$ inherits the port numbering from the parent edges.

5. Assuming that the graph $\mathcal{G}$ has a max-min LP associated with it, the max-min LP associated with the unfolding $\mathcal{G}'$ is defined by inheritance from the parent nodes and edges. In particular, the type of each node (agent, constraint, objective) is the type of the parent node, and the coefficients associated with the edges (a_{iv}, c_{kv}) are inherited from the parent edges.

6. Any two nodes of $\mathcal{G}'$ with the same parent are related by an automorphism of $\mathcal{G}'$. In particular, any deterministic local algorithm in the port numbering model must give the same output on any two nodes with the same parent. Any locally computed feasible solution of the max-min LP associated with $\mathcal{G}'$ defines a feasible solution of the max-min LP associated with $\mathcal{G}$, with the same utility.

7. Any feasible solution of the max-min LP associated with $\mathcal{G}$ defines, by inheritance, a feasible solution of the max-min LP associated with $\mathcal{G}'$, with the same utility.

8. A locally computed feasible solution of $\mathcal{G}'$ with utility at least $1/\alpha$ times the utility of any feasible solution of $\mathcal{G}'$ yields an α-approximation of the optimum of $\mathcal{G}$.

4. LOCAL TRANSFORMATIONS

Consider an arbitrary max-min LP associated with the graph $\mathcal{G}$. In this section we carry out a sequence of locally computable transformations, with the goal of arriving at a more structured max-min LP. The transformations are applied in the order of presentation, from §4.2 to §4.6. We describe each individual transformation in three parts:

1. A description of the transformation.

2. Mapping a solution of the transformed instance back to the original instance.

3. Implications to approximability. We write $\omega(\cdot)$ for the utility of the original instance and $\omega'(\cdot)$ for the utility of the transformed instance.

Figure 2 illustrates the transformations that modify the communication graph $\mathcal{G}$.

To avoid degenerate cases, we assume that each constraint and objective is adjacent to at least one agent, and every agent is adjacent to at least one constraint and at least one objective, that is, $|V_i| \geq 1$, $|V_k| \geq 1$, $|K_v| \geq 1$, and $|I_v| \geq 1$ for all $i \in I$, $k \in K$, $v \in V$. Indeed, isolated constraints can be deleted, isolated objectives force the optimum of (2) to zero, non-contributing agents can be set to zero, and unconstrained agents can be set to $+\infty$. Furthermore, we assume that $\mathcal{G}$ is connected, as we can handle each connected component independently.

4.1 Implementation details

Even though a description of a local algorithm often involves interleaved steps of communication and computation, it should be noted that any local algorithm with local horizon D can always be implemented as follows:

1. Each node gathers full information about its radius D neighbourhood; this is the *local view* of the agent.

2. Each node simulates the algorithm in its local view to determine its output.

As pointed out in §3, we can assume that the local view is a tree. Furthermore, in our case it is sufficient that each *agent* performs these steps – constraints and objectives do not need to produce any output. Therefore we can implement each transformation presented in this section as follows (with a small increase of the local horizon):

1. Each agent gathers its local view, up to some constant distance. This is a tree.

2. Each agent performs the transformation in its local view. The result is a graph, possibly with cycles.

3. Each agent unfolds the graph to obtain a tree, discarding parts that are beyond its local horizon.

4. Each agent simulates the rest of the local algorithm in this tree, and applies the back-mapping to determine its output.

In some of the transformations, new agent nodes are created. In §4.2, the output of the new agents is not needed. In §4.4 and §4.5, existing agents can simulate their copies and compute the back-mapping.

The nontrivial part is to make sure that the transformations can be performed deterministically: if the local views of agents u and v partially overlap, and both agents perform a transformation in the common part, the common parts must be identical after the transformation. In particular, the port numbers must be identical. In the following, we show how to achieve this for each transformation.

4.2 Augmenting singleton constraints

After this transformation, $|V_i| \geq 2$ for each $i \in I$. In essence, we augment degree-1 constraints with a cycle that does not affect the original instance.

The transformation. For each constraint $i \in I$ with $|V_i| = 1$, introduce three new agents, s, t, and u, two new objectives, h and ℓ, and one new constraint, j. Let $v \in V_i$ be the original agent adjacent to i, and let $k \in K_v$ be an objective adjacent to v. Set

$$\begin{aligned} a_{is} &= a_{jt} = a_{ju} = 1, \\ c_{hs} &= c_{\ell s} = 1, \\ c_{ht} &= c_{\ell u} = 2\sum_{w \in V_k} c_{kw} \min_{i \in I_w} \frac{1}{a_{iw}}. \end{aligned}$$

Figure 2 shows the structure of the modified LP.

We can use an arbitrary, fixed port numbering within the cycle induced by $\{s, h, t, j, u, \ell\}$. The edge $\{i, s\}$ that joins the cycle and the original graph is the last edge both from the perspective of i and s.

Mapping back. Let $\mathbf{x}'$ be any feasible solution of the transformed instance. We map this back by setting $x_v = x'_v$ for all original agents $v \in V$.

Approximation ratio. Observe that we can always set $x'_s = 0$ and $x'_t = x'_u = 1/2$ without decreasing the objective value $\omega'(\mathbf{x}')$. Thus the optima of the original and transformed instances coincide, and any approximation ratio is preserved.

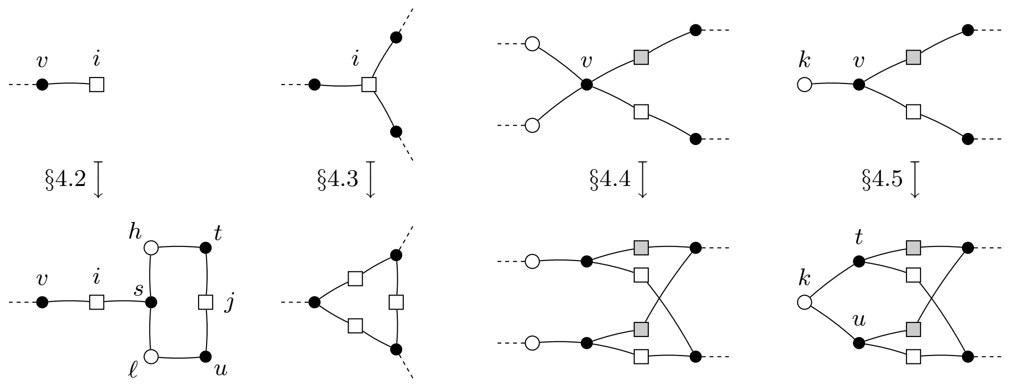

Figure 2: Local transformations in §4.2–§4.5.

4.3 Reducing the degree of constraints

After this transformation, $|V_i| = 2$ for each $i \in I$. This transformation is from prior work [6].

The transformation. Replace each constraint $i \in I$ with $|V_i| > 2$ by the $\binom{|V_i|}{2}$ constraints

$$a_{iu}x_u + a_{iv}x_v \le 1, \quad \forall\, u, v \in V_i,\ u < v. \tag{3}$$

See Figure 2 for an illustration in the case $|V_i| = 3$.

To see that the port numbers can be chosen deterministically, consider an agent $v \in V_i$. From the perspective of v, the edge $\{v, i\}$ in the original graph is replaced by $|V_i| - 1$ edges, and there is a one-to-one mapping between the new edges and the agents $V_i \setminus \{v\}$. Since we can order the set V_i by using the port numbering in i, we can also order the new edges. Each new constraint can also use the ordering of V_i to choose its port numbers.

Mapping back. Let $\mathbf{x}'$ be an arbitrary feasible solution of the transformed instance. We map this back to a feasible solution $\mathbf{x}$ of the original instance by setting

$$x_v = \frac{2x'_v}{\max_{i\in I_v} |V_i|}, \quad \forall\, v \in V. \tag{4}$$

To verify that $\mathbf{x}$ is feasible, consider an arbitrary original constraint $i \in I$. By the previous transformation, we have $|V_i| \ge 2$. Taking the sum over all the constraints (3) replacing i, or, if $|V_i| = 2$, considering the original constraint, we have

$$\sum_{v\in V_i} \big(|V_i| - 1\big) a_{iv}x'_v \le \frac{|V_i|\big(|V_i| - 1\big)}{2}.$$

By (4) we thus have

$$\sum_{v\in V_i} a_{iv}x_v \le \sum_{v\in V_i} \frac{2a_{iv}x'_v}{|V_i|} \le 1.$$

Approximation ratio. Since the objectives are unchanged in the transformation, we have

$$\omega(\mathbf{x}) \ge \frac{2\omega'(\mathbf{x}')}{\Delta_I}$$

by linearity and (4). Furthermore, an optimal solution of the original instance is a feasible solution of the transformed instance. Therefore, if $\mathbf{x}'$ is an α-approximate solution, then $\mathbf{x}$ is an $\alpha\Delta_I/2$-approximate solution.

4.4 Associating a unique objective with each agent

After this transformation, $|K_v| = 1$ for each $v \in V$.

The transformation. For each agent $v \in V$ with $|K_v| > 1$, replace v with $|K_v|$ copies of v as follows. Associate each copy of v with a unique objective in K_v. Replace each constraint adjacent to v with $|K_v|$ copies of the constraint, with v replaced by a unique copy in each constraint; see Figure 2. The coefficients are unchanged.

To choose the port numbers, consider a node $u \ne v$ adjacent to an $i \in I_v$. From the perspective of u, the edge $\{u, i\}$ is replaced by $|K_v|$ edges, and there is a one-to-one mapping between the new edges and the objectives in K_v; furthermore, the port numbering in v imposes an ordering on K_v. All other cases are straightforward; for example, each copy of a constraint simply inherits the port numbering from the original graph.

Mapping back. Let $\mathbf{x}'$ be a feasible solution of the transformed instance. By symmetry we can assume that all copies u of v have the same value x'_u without decreasing the objective value $\omega'(\mathbf{x}')$; indeed, if the values are different, just set all copies to the maximum among these values. Mapping back is done simply by identifying the copies back to the original.

Approximation ratio. Preserved. The optima of the original and the transformed instance coincide.

4.5 Augmenting singleton objectives

After this transformation, $|V_k| \ge 2$ for each $k \in K$.

The transformation. For each objective $k \in K$ with $|V_k| = 1$, let v be the unique agent adjacent to k. Replace v with two copies, t and u, and replace each constraint adjacent to v with two copies of the constraint, one containing t in place of v, and the other containing u in place of v. Let $c_{kt} = c_{ku} = c_{kv}/2$. The coefficients are otherwise unchanged.

To choose the port numbers, the new edges that point to t and its adjacent constraints are ordered before the new edges that point to u and its adjacent constraints.

Mapping back. Let $\mathbf{x}'$ be a feasible solution of the transformed instance. By symmetry we can assume that the copies of v have the same value $x'_t = x'_u$ without decreasing the objective value $\omega'(\mathbf{x}')$; indeed, if the values are different, just set all copies to the maximum among these values. Mapping back is done simply by identifying the copies back to the original.

Approximation ratio. Preserved. The optima of the original and the transformed instance coincide.

4.6 Normalising coefficients

After this transformation, $c_{kv} = 1$ for each $\{k, v\} \in E$ with $k \in K$ and $v \in V$.

The transformation. For each $v \in V$, let $k(v)$ be the unique objective in K_v. For each $v \in V$, $i \in I$, and $k \in K$, divide a_{iv} and c_{kv} by $c_{k(v)v}$. The graph is not changed, and port numbering is preserved.

Mapping back. For each $v \in V$, multiply x_v by $c_{k(v)v}$.

Approximation ratio. Preserved.

5. LOCAL ALGORITHM

Throughout this section we consider a max-min LP associated with a bipartite graph $\mathcal{G} = (V \cup I \cup K, E)$ with these properties that follow from §3 and §4:

– the graph $\mathcal{G}$ is an unfolding of a finite graph,

– $|K_v| = 1$ and $|I_v| \geq 1$ for each agent $v \in V$,

– $|V_i| = 2$ for each constraint $i \in I$,

– $|V_k| \geq 2$ for each objective $k \in K$,

– $c_{kv} = 1$ for each pair of adjacent $k \in K$ and $v \in V$.

It follows that $\mathcal{G}$ is countably infinite.

Recall that $k(v)$ is the unique objective adjacent to $v \in V$. Let $N(v) = V_{k(v)} \setminus \{v\}$ be the set of other agents adjacent to this objective. For a constraint $i \in I$ and an agent $v \in V_i$, denote by $n(v, i)$ the unique agent other than v in V_i. See Figure 1 for an illustration.

Let $R = 2, 3, \ldots$ be a fixed parameter that will determine the local horizon and the approximation ratio. Let $r = R-2$.

5.1 An upper bound via alternating trees

A walk $W = (u_0, e_1, u_1, \ldots, e_\ell, u_\ell)$ in $\mathcal{G}$ is *alternating* if (i) for all $1 \leq j < j' \leq \ell$ with $u_j \in K$ and $u_{j'} \in K$, there exists a $j < j'' < j'$ with $u_{j''} \in I$; and (ii) for all $1 \leq j < j' \leq \ell$ with $u_j \in I$ and $u_{j'} \in I$, there exists a $j < j'' < j'$ with $u_{j''} \in K$.

Let $u \in V$ be an arbitrary agent, and consider the subgraph $\mathcal{A}_u$ of $\mathcal{G}$ induced by the nodes reachable via alternating paths starting at u that (i) traverse the constraint $k(u)$ and have length at most $4r+3$; or (ii) have length at most 1. The *level* of a node of $\mathcal{A}_u$ is its distance to $k(u)$, with the exception of u, which we define to have level -1, and the constraints adjacent to u, which we define to have level -2. See Figure 1 for an illustration.

Associate with $\mathcal{A}_u$ a max-min LP by restriction from the max-min LP associated with $\mathcal{G}$.

LEMMA 1. *The graph $\mathcal{A}_u$ is a finite tree. Moreover, every objective in $\mathcal{A}_u$ is at level* $0 \pmod 4$*, every agent in $\mathcal{A}_u$ is at level either* $1 \pmod 4$ *or* $3 \pmod 4$*, and every constraint in $\mathcal{A}_u$ is at level* $2 \pmod 4$*. The leaves of $\mathcal{A}_u$ are constraints at levels -2 and $4r+2$. For any objective k in $\mathcal{A}_u$ and any agent v adjacent to k in $\mathcal{G}$, the agent v occurs in $\mathcal{A}_u$ and is adjacent to k in $\mathcal{A}_u$.*

PROOF. By induction on R using the assumptions on the structure of $\mathcal{G}$. □

LEMMA 2. *The optimum value of the max-min LP associated with $\mathcal{A}_u$ is an upper bound on the value of any feasible solution of the max-min LP associated with $\mathcal{G}$.*

PROOF. Because $\mathcal{A}_u$ is finite, the optimum is well defined. It suffices to show that any feasible solution of $\mathcal{G}$ is (by restriction) a feasible solution of $\mathcal{A}_u$. This follows from Lemma 1: the objectives in $\mathcal{A}_u$ are identical to those in $\mathcal{G}$, and the constraints on variables in $\mathcal{A}_u$ are either identical or relaxed from $\mathcal{G}$ (at leaves or non-alternating constraints). □

5.2 The optimum of $\mathcal{A}_u$

The tree structure of $\mathcal{A}_u$ enables a recursive characterisation of the optimum that proceeds level-wise towards u. Denote by $L(u, \ell)$ the set of all nodes at level ℓ in $\mathcal{A}_u$.

Define

$$f^{+}_{u,v,0}(\omega) = \min_{i \in I_v} \frac{1}{a_{iv}} \tag{5}$$

for each $v \in L(u, 4r+1)$,

$$f^{-}_{u,v,d}(\omega) = \max\Big\{0,\, \omega - \sum_{w \in N(v)} f^{+}_{u,w,d}(\omega)\Big\} \tag{6}$$

for each $d = 0, 1, \ldots, r$ and $v \in L(u, 4(r-d)-1)$, and

$$f^{+}_{u,v,d}(\omega) = \min_{i \in I_v} \frac{1 - a_{i,n(v,i)} f^{-}_{u,n(v,i),d-1}(\omega)}{a_{iv}} \tag{7}$$

for each $d = 0, 1, \ldots, r$ and $v \in L(u, 4(r-d)+1)$.

Example 1. Figure 1 illustrates the computation of (5)–(7) in the case $r = 1$, for a given ω. The agents at level 5 in the tree $\mathcal{A}_u$ choose the *largest* values $f^{+}_{u,\cdot,0}(\omega)$ that do not violate any of the constraints at level 6. The agents at level 3 choose the *smallest* values $f^{-}_{u,\cdot,0}(\omega)$ so that the objectives at level 4 meet the requirement ω. The agents at level 1 choose the *largest* values $f^{+}_{u,\cdot,1}(\omega)$ that do not violate any of the constraints at level 2, and the agents at level -1 choose the *smallest* values $f^{-}_{u,\cdot,1}(\omega)$ so that the objectives at level 0 meet the requirement ω. If we interpret f_u as a candidate solution to the max-min LP associated with $\mathcal{A}_u$, we notice that the utility of the solution would be at least ω, but the solution is not necessarily feasible. In particular, the values f^{+}_u might be negative and the constraints at level -2 might be violated.

Now let t_u be the maximum value $\omega \geq 0$ such that

$$f^{+}_{u,v,d}(\omega) \geq 0 \tag{8}$$

for all $0 \leq d \leq r$ and $v \in L(u, 4(r-d)+1)$, and

$$f^{-}_{u,u,r}(\omega) \leq \min_{i \in I_u} \frac{1}{a_{iu}}. \tag{9}$$

Note that the maximum exists because (6) and (7) can be expressed using linear inequalities (by introducing additional variables), and $\omega = 0$ is a feasible value. Indeed, as the following lemma shows, the values $f^{\pm}_{u,v,d}(t_u)$ are simply a specific optimum solution of the LP associated with the tree $\mathcal{A}_u$; in essence, we focus on a particular extreme point among all optimal solutions of the LP.

LEMMA 3. *Let* $\mathbf{x}$ *be any feasible solution that achieves the objective value* ω *in the max-min LP associated with* $\mathcal{A}_u$. *Then*

$$0 \le x_v \le f^+_{u,v,d}(\omega) \tag{10}$$

for all $d = 0, 1, \ldots, r$ *and* $v \in L(u, 4(r-d)+1)$, *and*

$$f^-_{u,v,d}(\omega) \le x_v \le \min_{i \in I_v} \frac{1}{a_{iv}} \tag{11}$$

for all $d = 0, 1, \ldots, r$ *and* $v \in L(u, 4(r-d)-1)$. *In particular,* t_u *is the optimum utility of the max-min LP associated with* $\mathcal{A}_u$.

PROOF. By induction on d and in order of evaluation of the recursive steps (6) and (7).

To set up the base case at $d = 0$, observe that by the feasibility of $\mathbf{x}$ and (5) we have

$$x_v \le \min_{i \in I_v} \frac{1}{a_{iv}} = f^+_{u,v,0}(\omega), \quad \forall\, v \in L(u, 4r+1).$$

Next consider any $0 \le d \le r$ and assume inductively that

$$x_w \le f^+_{u,w,d}(\omega), \quad \forall\, w \in L\big(u, 4(r-d)+1\big).$$

Consider an arbitrary $v \in L(u, 4(r-d)-1)$. Observe that for all $w \in V_{k(v)}$ it holds that either $v = w$ or $w \in L(u, 4(r-d)+1)$. If we have

$$0 \ge \omega - \sum_{w \in N(v)} f^+_{u,w,d}(\omega),$$

then (6) implies $f^-_{u,v,d}(\omega) = 0 \le x_v$; otherwise (6) implies

$$f^-_{u,v,d}(\omega) = \omega - \sum_{w \in N(v)} f^+_{u,w,d}(\omega) \le \omega - \sum_{w \in N(v)} x_w \le x_v.$$

Here the first inequality follows by the induction hypothesis, and the second inequality follows by assumption that $\mathbf{x}$ achieves the objective value ω; in particular,

$$\sum_{w \in V_{k(v)}} x_w = x_v + \sum_{w \in N(v)} x_w \ge \omega.$$

To complete the induction, consider any $1 \le d \le r$ and assume inductively that

$$f^-_{u,w,d-1}(\omega) \le x_w, \quad \forall\, w \in L\big(u, 4(r-(d-1))-1\big).$$

Consider an arbitrary $v \in L(u, 4(r-d)+1)$. Observe that $n(v,i) \in L(u, 4(r-(d-1))-1)$. Because $\mathbf{x}$ is feasible, we have

$$a_{iv}x_v + a_{i,n(v,i)}x_{n(v,i)} \le 1$$

for all $i \in I_v$. Thus, the inductive hypothesis and (7) imply

$$\begin{aligned} x_v &\le \min_{i \in I_v} \frac{1 - a_{i,n(v,i)}x_{n(v,i)}}{a_{iv}} \\ &\le \min_{i \in I_v} \frac{1 - a_{i,n(v,i)}f^-_{u,n(v,i),d-1}(\omega)}{a_{iv}} = f^+_{u,v,d}(\omega). \end{aligned}$$

To conclude that t_u is the optimum utility of the max-min LP associated with $\mathcal{A}_u$, consider an optimal solution $\mathbf{x}$ with the objective value ω^*. Observe that (10) and (11) imply (8) and (9). Therefore $t_u \ge \omega^*$. Furthermore, $t_u > \omega^*$ would contradict the assumption that $\mathbf{x}$ is optimal. $\square$

In what follows we use the shorthand notation

$$\begin{aligned} f^+_{u,v,d} &= f^+_{u,v,d}(t_u), \\ f^-_{u,v,d} &= f^-_{u,v,d}(t_u). \end{aligned}$$

The values $f^\pm_{u,v,d}$ are used in the analysis; the local algorithm needs to know only the optimums t_u. To keep our analysis simpler, we assume here that the node u uses an LP solver to find the optimum of the LP associated with $\mathcal{A}_u$, and hence the exact value of t_u. In a practical implementation of our algorithm, we do not need to invoke an LP solver; a simple binary search for an approximation of t_u is sufficient.

5.3 Smoothing

For each agent $v \in V$ in $\mathcal{G}$, let s_v be the minimum of the values t_u over all agents $u \in V$ at distance at most $4r+2$ from v in $\mathcal{G}$. For all $v \in V$ in $\mathcal{G}$ and all $d = 0, 1, \ldots, r$, define

$$g^+_{v,0} = \min_{i \in I_v} \frac{1}{a_{iv}}, \tag{12}$$

$$g^-_{v,d} = \max\Big\{0,\, s_v - \sum_{w \in N(v)} g^+_{w,d}\Big\}, \tag{13}$$

$$g^+_{v,d} = \min_{i \in I_v} \frac{1 - a_{i,n(v,i)}g^-_{n(v,i),d-1}}{a_{iv}}, \quad d \ge 1. \tag{14}$$

Example 2. We can interpret (12)–(14) in terms of the functions $f^\pm$ defined in (5)–(7). For example, consider the agent b in Figure 1. Let us determine the value $g^-_{b,0}$. Choose any agent u (for example, the one shown in the figure) such that b is a level-3 node in the tree $\mathcal{A}_u$, that is, the function $f^-_{u,b,0}(\omega)$ is defined. Then we have $g^-_{b,0} = f^-_{u,b,0}(s_b)$, independent of the choice of the agent u.

LEMMA 4. *For all* $u \in V$ *and all* $d = 0, 1, \ldots, r$ *it holds that*

$$g^-_{v,d} \le f^-_{u,v,d}, \quad \forall\, v \in L\big(u, 4(r-d)-1\big), \tag{15}$$

$$f^+_{u,v,d} \le g^+_{v,d}, \quad \forall\, v \in L\big(u, 4(r-d)+1\big). \tag{16}$$

PROOF. Consider an arbitrary $u \in V$. For all $w \in V$ at distance at most $4r+2$ from u we have, by the definition of s_w,

$$0 \le s_w \le t_u. \tag{17}$$

We proceed by induction on d. To establish the base case at $d = 0$, observe by (5) and (12) that $f^+_{u,v,0} = g^+_{v,0}$ for all $v \in L(u, 4r+1)$.

Next consider any $0 \le d \le r$ and assume inductively that

$$f^+_{u,w,d} \le g^+_{w,d}, \quad \forall\, w \in L\big(u, 4(r-d)+1\big).$$

Consider an arbitrary $v \in L(u, 4(r-d)-1)$. Observe that for all $w \in V_{k(v)}$ either $w = v$ or $w \in L(u, 4(r-d)+1)$. Apply (13), (17), the inductive hypothesis, and (6) to obtain

$$\begin{aligned} g^-_{v,d} &= \max\Big\{0,\, s_v - \sum_{w \in N(v)} g^+_{w,d}\Big\} \\ &\le \max\Big\{0,\, t_u - \sum_{w \in N(v)} f^+_{u,w,d}\Big\} = f^-_{u,v,d}. \end{aligned}$$

Finally, consider any $1 \le d \le r$ and assume inductively that

$$g^-_{w,d-1} \le f^-_{u,w,d-1}, \quad \forall\, w \in L\big(u, 4(r-(d-1))-1\big).$$

Consider an arbitrary $v \in L(u, 4(r-d)+1)$. Observe that for all $i \in I_v$ it holds that $n(v,i) \in L(u, 4(r-(d-1))-1)$. Apply (7), the inductive hypothesis, and (14), to obtain

$$\begin{aligned} f^{+}_{u,v,d} &= \min_{i \in I_v} \frac{1 - a_{i,n(v,i)} f^{-}_{u,n(v,i),d-1}}{a_{iv}} \\ &\le \min_{i \in I_v} \frac{1 - a_{i,n(v,i)} g^{-}_{n(v,i),d-1}}{a_{iv}} = g^{+}_{v,d}. \end{aligned}$$

The induction is now complete. □

LEMMA 5. *For all $v \in V$ it holds that*

$$\begin{aligned} g^{+}_{v,r} &\ge 0, \\ g^{-}_{v,r} &\le \min_{i \in I_v} \frac{1}{a_{iv}}. \end{aligned}$$

PROOF. Let $v \in V$ be arbitrary and choose $u \in N(v)$; such a u exists because every objective is adjacent to at least two agents. We have

$$g^{+}_{v,r} \ge f^{+}_{u,v,r} \ge 0$$

by (16) and (8). Similarly, let $u = v$ to obtain

$$g^{-}_{v,r} \le f^{-}_{v,v,r} \le \min_{i \in I_v} \frac{1}{a_{iv}}$$

by (15) and (9). □

LEMMA 6. *For all $v \in V$ and $d = 1, 2, \dots, r$ it holds that $g^{-}_{v,d-1} \le g^{-}_{v,d}$ and $g^{+}_{v,d} \le g^{+}_{v,d-1}$.*

PROOF. By induction on d. To set up the base case at $d = 1$, observe first that $g^{-}_{v,0} \ge 0$ by (13). By (14) and (12) thus

$$\begin{aligned} g^{+}_{v,1} &= \min_{i \in I_v} \frac{1 - a_{i,n(v,i)} g^{-}_{n(v,i),0}}{a_{iv}} \\ &\le \min_{i \in I_v} \frac{1}{a_{iv}} = g^{+}_{v,0}. \end{aligned}$$

Next consider any $1 \le d \le r$ and assume inductively that $g^{+}_{v,d} \le g^{+}_{v,d-1}$. Apply (13) and the inductive hypothesis to obtain

$$\begin{aligned} g^{-}_{v,d} &= \max\Big\{0,\, s_v - \sum_{w \in N(v)} g^{+}_{w,d}\Big\} \\ &\ge \max\Big\{0,\, s_v - \sum_{w \in N(v)} g^{+}_{w,d-1}\Big\} = g^{-}_{v,d-1}. \end{aligned}$$

Finally, consider any $2 \le d \le r$ and assume inductively that $g^{-}_{v,d-2} \le g^{-}_{v,d-1}$. Apply (14) and the inductive hypothesis to obtain

$$\begin{aligned} g^{+}_{v,d} &= \min_{i \in I_v} \frac{1 - a_{i,n(v,i)} g^{-}_{n(v,i),d-1}}{a_{iv}} \\ &\le \min_{i \in I_v} \frac{1 - a_{i,n(v,i)} g^{-}_{n(v,i),d-2}}{a_{iv}} = g^{+}_{v,d-1}. \end{aligned}$$

This completes the induction. □

LEMMA 7. *For all $v \in V$ and $d = 0, 1, \dots, r$ it holds that $g^{+}_{v,d} \ge 0$.*

PROOF. By Lemmata 5 and 6. □

Finally, each agent v outputs the value

$$x_v = \frac{1}{2R} \sum_{d=0}^{r} \big(g^{+}_{v,d} + g^{-}_{v,d}\big). \tag{18}$$

This completes the description of the algorithm. The algorithm is local, with the local horizon $\Theta(R)$. We proceed to show that the vector $\mathbf{x}$ is a feasible solution, and within factor $2(1 - 1/\Delta_K) + \epsilon'$ of the optimum.

6. ANALYSIS

We start by partitioning the set of agents V into *up-agents* and *down-agents* such that (i) every constraint is adjacent to exactly one up-agent and exactly one down-agent; and (ii) every objective is adjacent to exactly one up-agent. Figure 1 shows an example of such a partition; note that if the node u is designated as an up-agent, then there is only one way to choose the types of the other agents in the tree $\mathcal{A}_u$.

Associate an integer *layer* to each node of $\mathcal{G}$ as follows. First, fix an arbitrary objective $k \in K$ to be at layer 0. Then, determine the layer of every other node u by considering the unique directed path connecting k to u. The layer of u is determined by taking the sum of the weights of the directed edges in the path, where the weights are displayed in Figure 3. See Figure 1 for an example.

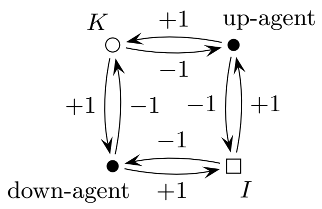

Figure 3: The weights used to assign the layers.

LEMMA 8. *The layers of the nodes of $\mathcal{G}$ satisfy the following four properties: every objective has layer* 0 (mod 4)*, every down-agent has layer* 1 (mod 4)*, every constraint has layer* 2 (mod 4)*, and every up-agent has layer* 3 (mod 4)*.*

PROOF. Immediate from Figure 3. □

6.1 Shifting strategy

Let $j = 0, 1, \dots, R-1$ be a shift parameter. For each agent $v \in V$, represent the layer of v uniquely as $4(Rc+j)+4d+e$ for integers c, d, e with $0 \le d \le R-1$ and $e \in \{-1, 1\}$. Recall that $r = R-2$. Associate with v the value

$$y_v(j) = \begin{cases} 0 & \text{if } d = R-1, \\ g^{-}_{v,r-d} & \text{if } d \le R-2 \text{ and } e = -1, \\ g^{+}_{v,r-d} & \text{if } d \le R-2 \text{ and } e = 1. \end{cases} \tag{19}$$

Observe that a down-agent has $e = 1$ and an up-agent has $e = -1$, regardless of j.

For an objective k and a vector $\mathbf{z}$ indexed by the agents, let

$$\omega_k(\mathbf{z}) = \sum_{v \in V_k} z_v.$$

LEMMA 9. *The vector $\mathbf{y}(j)$ is a feasible solution of the max-min LP associated with $\mathcal{G}$. For every objective $k \in K$, it holds that*

$$\begin{aligned} \omega_k\big(\mathbf{y}(j)\big) &= 0 && \text{if } k \text{ is at layer } 4j-4 \pmod{4R}, \\ \omega_k\big(\mathbf{y}(j)\big) &\geq \min_{v \in V_k} s_v && \text{otherwise.} \end{aligned}$$

PROOF. *Feasibility.* The vector is nonnegative by (13) and Lemma 7. Consider an arbitrary constraint $i \in I$. By Lemma 8 we can represent the layer of i uniquely as

$$4(Rc+j) + 4d + 2$$

for integers c, d with $0 \leq d \leq R-1$. Let $V_i = \{v, w\}$ with v at layer $4(Rc+j)+4d+1$ and w at layer $4(Rc+j)+4(d+1)-1$.

First consider the case $d = R-1$. Note that the layer of w is actually $4(R(c+1)+j)-1$, that is, $d=0$ and $e=-1$ for w. By (19) and Lemma 5, we have

$$a_{iv} y_v(j) + a_{iw} y_w(j) \;=\; a_{iw} g^-_{w,r} \;\leq\; 1.$$

Next consider the case $d = R-2$. By (19) and (12), we have

$$a_{iv} y_v(j) + a_{iw} y_w(j) \;=\; a_{iv} g^+_{v,0} \;\leq\; 1.$$

Finally consider the case $d < R-2$. By (19) and (14), we have

$$\begin{aligned} &a_{iv} y_v(j) + a_{iw} y_w(j) \\ &\quad= a_{iv} g^+_{v,r-d} + a_{iw} g^-_{w,r-d-1} \\ &\quad\leq a_{iv} \frac{1 - a_{iw} g^-_{w,r-d-1}}{a_{iv}} + a_{iw} g^-_{w,r-d-1} \;=\; 1. \end{aligned}$$

The claim follows since i was arbitrary.

Objectives. Consider an arbitrary objective k in $\mathcal{G}$. By Lemma 8 we can represent the layer of k uniquely as

$$4(Rc+j) + 4d$$

for integers c, d with $0 \leq d \leq R-1$. There is a unique up-agent in V_k at layer $4(Rc+j)+4d-1$. Denote this agent by v. The other agents $w \in N(v)$ are down-agents at layer $4(Rc+j)+4d+1$.

First consider the case $d = R-1$. By (19) we have

$$\omega_k\big(\mathbf{y}(j)\big) \;=\; y_v(j) + \sum_{w \in N(v)} y_w(j) \;=\; 0.$$

Then consider the case $d \leq R-2$. By (19) and (13), we have

$$\begin{aligned} \omega_k\big(\mathbf{y}(j)\big) &= y_v(j) + \sum_{w \in N(v)} y_w(j) \\ &= g^-_{v,r-d} + \sum_{w \in N(v)} g^+_{w,r-d} \\ &\geq s_v - \sum_{w \in N(v)} g^+_{w,r-d} + \sum_{w \in N(v)} g^+_{w,r-d} \\ &= s_v \;\geq\; \min_{u \in V_k} s_u. \end{aligned}$$

The claim follows because k was arbitrary. $\square$

Let us now average over all values of the shift parameter $j = 0, 1, \ldots, R-1$ to obtain

$$\begin{aligned} y_v &= \frac{1}{R} \sum_{j=0}^{R-1} y_v(j) \\ &= \begin{cases} \dfrac{1}{R} \displaystyle\sum_{d=0}^{r} g^-_{v,d} & \text{if } v \text{ is an up-agent,} \\ \dfrac{1}{R} \displaystyle\sum_{d=0}^{r} g^+_{v,d} & \text{if } v \text{ is a down-agent.} \end{cases} \end{aligned} \tag{20}$$

LEMMA 10. *The vector $\mathbf{y}$ is a feasible solution of the max-min LP associated with $\mathcal{G}$. For every objective $k \in K$, it holds that*

$$\omega_k(\mathbf{y}) \;\geq\; \left(1 - \frac{1}{R}\right) \min_{v \in V_k} s_v.$$

PROOF. Follows from Lemma 9. $\square$

6.2 Averaging

Associate with each agent $v \in V$ a solution $\mathbf{y}^{\uparrow v}$ defined as follows. Choose the layers so that v is an up-agent; this is always possible. Let $\mathbf{y}^{\uparrow v}$ be the value of (20).

LEMMA 11. *The vector $\mathbf{x}$ is a feasible solution of the max-min LP associated with $\mathcal{G}$.*

PROOF. Consider an arbitrary constraint $i \in I$; let $V_i = \{v, w\}$. Note that whenever v is an up-agent w is a down-agent and vice versa. Let

$$\mathbf{z} \;=\; \frac{\mathbf{y}^{\uparrow v} + \mathbf{y}^{\uparrow w}}{2}.$$

By (20) and (18) we have

$$z_v \;=\; \frac{y_v^{\uparrow v} + y_v^{\uparrow w}}{2} \;=\; \frac{1}{2}\left(\frac{1}{R}\sum_{d=0}^{r} g^-_{v,d} + \frac{1}{R}\sum_{d=0}^{r} g^+_{v,d}\right) \;=\; x_v$$

and $z_w = x_w$. By Lemma 10, the solutions $\mathbf{y}^{\uparrow v}$ and $\mathbf{y}^{\uparrow w}$ do not violate the constraint i. Therefore

$$\begin{aligned} a_v x_v + a_w x_w &= a_v z_v + a_w z_w \\ &= \frac{(a_v y_v^{\uparrow v} + a_w y_w^{\uparrow v}) + (a_v y_v^{\uparrow w} + a_w y_w^{\uparrow w})}{2} \\ &\leq \frac{1+1}{2} = 1. \end{aligned}$$

We conclude that the solution $\mathbf{x}$ does not violate the constraint i. $\square$

LEMMA 12. *For every objective $k \in K$,*

$$\omega_k(\mathbf{x}) \;\geq\; \frac{1}{2}\left(1 - \frac{1}{R}\right) \frac{|V_k|}{|V_k| - 1} \min_{v \in V_k} s_v. \tag{21}$$

PROOF. Consider an arbitrary objective k in $\mathcal{G}$. Note that whenever $v \in V_k$ is an up-agent, each $w \in N(v)$ is a down-agent. Let

$$\mathbf{z} \;=\; \frac{1}{|V_k|} \sum_{v \in V_k} \mathbf{y}^{\uparrow v}.$$

By (20) and (18) we have

$$\begin{aligned}\frac{|V_k|}{2(|V_k|-1)} z_u &= \frac{1}{2(|V_k|-1)} \sum_{v \in V_k} y_u^{\uparrow v} \\ &= \frac{1}{|V_k|-1} \left(\frac{1}{2R} \sum_{d=0}^{r} g_{u,d}^{-} + \frac{|V_k|-1}{2R} \sum_{d=0}^{r} g_{u,d}^{+} \right) \\ &\leq x_u\end{aligned}$$

for all $u \in V_k$. By Lemma 10,

$$\begin{aligned}\omega_k(\mathbf{x}) &= \sum_{u \in V_k} x_u \geq \sum_{u \in V_k} \frac{|V_k|}{2(|V_k|-1)} z_u \\ &\geq \frac{|V_k|}{2(|V_k|-1)} \left(1 - \frac{1}{R}\right) \min_{v \in V_k} s_v .\end{aligned}$$

The claim follows. □

6.3 Completing the analysis

Lemmata 2 and 3 show that for any $v \in V$, the value t_v is an upper bound for the utility of any feasible solution of the max-min LP instance associated with $\mathcal{G}$, and so is s_v. Lemma 12 therefore shows that our local algorithm achieves the approximation ratio of $2(1 - 1/\Delta_K)(1 + 1/(R-1))$, for the special case studied in §5.

Together with the local transformations of §4, taking into account the increase of the approximation in §4.3, we conclude that the max-min LP problem admits a local algorithm with the approximation ratio of

$$\Delta_I \left(1 - \frac{1}{\Delta_K}\right)\left(1 + \frac{1}{R-1}\right);$$

the local horizon is $\Theta(R)$. Theorem 1 follows by choosing a sufficiently large R.

The constants Δ_I and Δ_K are used solely in the analysis. The algorithm is oblivious to these constants; only the value of R needs to be fixed in advance to run the algorithm. Naturally if one is given a specific ϵ, then the choice of the correct R requires information on Δ_I and Δ_K.

7. ACKNOWLEDGMENTS

We thank anonymous reviewers for their helpful suggestions. This research was supported in part by the Academy of Finland, Grants 116547 and 117499, by Helsinki Graduate School in Computer Science and Engineering (Hecse), and by the Foundation of Nokia Corporation.

8. REFERENCES

[1] D. Angluin. Local and global properties in networks of processors. In *Proc. 12th Symposium on Theory of Computing (STOC 1980)*, pages 82–93. ACM Press, 1980.

[2] M. Åstrand, P. Floréen, V. Polishchuk, J. Rybicki, J. Suomela, and J. Uitto. A local 2-approximation algorithm for the vertex cover problem, 2009. Manuscript submitted for publication.

[3] H. Attiya, M. Snir, and M. K. Warmuth. Computing on an anonymous ring. *J. ACM*, 35(4):845–875, 1988.

[4] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994.

[5] A. Czygrinow, M. Hańćkowiak, and W. Wawrzyniak. Fast distributed approximations in planar graphs. In *Proc. 22nd Symposium on Distributed Computing (DISC 2008)*, volume 5218 of *LNCS*, pages 78–92. Springer, 2008.

[6] P. Floréen, M. Hassinen, P. Kaski, and J. Suomela. Local approximation algorithms for a class of 0/1 max-min linear programs, 2008. Manuscript, arXiv:0806.0282 [cs.DC].

[7] P. Floréen, M. Hassinen, P. Kaski, and J. Suomela. Tight local approximation results for max-min linear programs. In *Proc. 4th Workshop on Algorithmic Aspects of Wireless Sensor Networks (Algosensors 2008)*, volume 5389 of *LNCS*, pages 2–17. Springer, 2008.

[8] P. Floréen, P. Kaski, T. Musto, and J. Suomela. Approximating max-min linear programs with local algorithms. In *Proc. 22nd International Parallel and Distributed Processing Symposium (IPDPS 2008)*. IEEE, 2008.

[9] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 32(1):130–136, 1985.

[10] F. Kuhn. *The Price of Locality: Exploring the Complexity of Distributed Coordination Primitives.* PhD thesis, ETH Zürich, 2005.

[11] F. Kuhn, T. Moscibroda, and R. Wattenhofer. The price of being near-sighted. In *Proc. 17th Symposium on Discrete Algorithms (SODA 2006)*, pages 980–989. ACM Press, 2006.

[12] C. Lenzen and R. Wattenhofer. Leveraging Linial's locality limit. In *Proc. 22nd Symposium on Distributed Computing (DISC 2008)*, volume 5218 of *LNCS*, pages 394–407. Springer, 2008.

[13] N. Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992.

[14] A. Mayer, M. Naor, and L. Stockmeryer. Local computations on static and dynamic graphs. In *Proc. 3rd Israel Symposium on the Theory of Computing and Systems (ISTCS 1995)*, pages 268–278. IEEE, 1995.

[15] M. Naor and L. Stockmeyer. What can be computed locally? *SIAM J. Comput.*, 24(6):1259–1277, 1995.

[16] C. H. Papadimitriou and M. Yannakakis. Linear programming without the matrix. In *Proc. 25th Symposium on Theory of Computing (STOC 1993)*, pages 121–129. ACM Press, 1993.

[17] J. Suomela. *Optimisation Problems in Wireless Sensor Networks: Local Algorithms and Local Graphs.* PhD thesis, University of Helsinki, Department of Computer Science, Helsinki, Finland, May 2009. To appear.

[18] J. Suomela. Survey of local algorithms. http://www.iki.fi/jukka.suomela/local-survey, 2009. Manuscript submitted for publication.

[19] M. Yamashita and T. Kameda. Computing on anonymous networks: Part I – characterizing the solvable cases. *IEEE Trans. Parallel Distrib. Systems*, 7(1):69–89, 1996.

[20] N. E. Young. Sequential and parallel algorithms for mixed packing and covering. In *Proc. 42nd Symposium on Foundations of Computer Science (FOCS 2001)*, pages 538–546. IEEE Computer Society Press, 2001.