A Unified Framework for Max-Min and Min-Max Fairness with Applications

Božidar Radunović and Jean-Yves Le Boudec
bozidar.radunovic@epfl.ch, jean-yves.leboudec@epfl.ch
EPFL, CH-1015 Lausanne, Switzerland *

July 12, 2002

Abstract

Max-min fairness is widely used in various areas of networking. In every case where it is used, there is a proof of existence and one or several algorithms for computing the max-min fair allocation; in most, but not all cases, they are based on the notion of bottlenecks. In spite of this wide applicability, there are still examples, arising in the context of mobile or peer-to-peer networks, where the existing theories do not seem to apply directly. In this paper, we give a unifying treatment of max-min fairness, which encompasses all existing results in a simplifying framework, and extends its applicability to new examples. First, we observe that the existence of max-min fairness is actually a geometric property of the set of feasible allocations (uniqueness always holds). There exist sets on which max-min fairness does not exist, and we describe a large class of sets on which a max-min fair allocation does exist. This class contains the compact, convex sets of $\mathbb{R}^N$, but not only. Second, we give a general purpose, centralized algorithm, called Max-min Programming, for computing the max-min fair allocation in all cases where it exists (whether the set of feasible allocations is in our class or not). Its complexity is of the order of N linear programming steps in $\mathbb{R}^N$, in the case where the feasible set is defined by linear constraints. We show that, if the set of feasible allocations has the free-disposal property, then Max-min Programming degenerates to a simpler algorithm, called Water Filling, whose complexity is much less. Free disposal corresponds to the cases where a bottleneck argument can be made, and Water Filling is the general form of all previously known centralized algorithms for such cases. Our derivations are based on the relation between max-min fairness and leximin ordering. All our results apply mutatis mutandis to min-max fairness. Our results apply to weighted, unweighted and util-max-min and min-max fairness. Distributed algorithms for the computation of max-min fair allocations are left outside the scope of this paper.

1 Introduction

Max-min fairness is a simple, well recognized approach to define fairness in networks [5]; it aims at allocating as much as possible to poor users, while not unnecessarily wasting resources (see later for a formal definition). It was used in window flow control protocols [7], then became very popular in the context of bandwidth sharing policies for ABR service in ATM networks [3]. It is now widely used

*The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

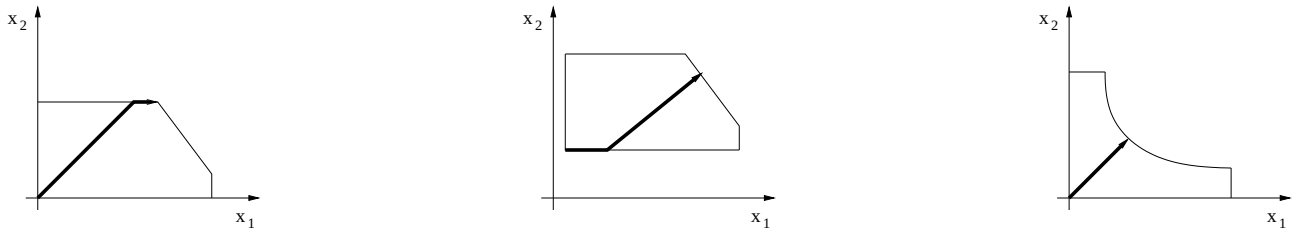

Figure 1: Various feasible sets for 2-dimensional examples. Left: with no minimal rate guarantee. Center: with minimal rate guarantee. Right: non-concave. The axes x_1 and x_2 represent rates allocated to two flows. The bold lines represent the steps of the Water-Filling algorithm.

in various areas of networking [17, 19, 18, 11, 9, 6, 15, 14, 7, 1]. Other concepts of fairness [8] are outside the scope of this paper.

One of the simplest max-min fairness examples, given in [5], is a single-path rate allocation. Suppose we have a network consisting of links with fixed capacities, and a set of source destination pairs that communicate over a single path each, and with fixed routing. The problem is to allocate a rate to each source-destination pair, while keeping the rate on each link below capacity. Here, we call a rate allocation max-min fair if one cannot increase the rate of a flow without decreasing the rate of an already smaller flow. A set of feasible rate allocations for a simple two source example is given on the left of fig 1. A definition dual to a max-min fair allocation is min-max fair allocation, and is used in the context of workload distribution, where the goal is to spread a given workload evenly to all the parties: see [13], where rates have to be allocated to available links as evenly as possible.

Max-min Fairness, Bottleneck and Water-Filling. Most of the existing works on max-min fairness rely on the notion of bottleneck link. Referring again to the single-path rate allocation example given above, we say that a link is a bottleneck for a given flow if the flow uses the link, if the link is fully utilized, and if the flow has the maximal rate among all the flows that use the link. It is shown in [5] for the above example that if each flow has a bottleneck link, then the rate allocation is max-min fair. An algorithm for obtaining a max-min fair rate allocation is also given: we increase rates of flows at the same pace, until a link is saturated. Then we fix the rates of flows passing through saturated link and keep increasing others. The procedure is repeated until there are no more flows whose rate can be increased. This algorithm is also known as a *water-filling* algorithm (WF). A simple example of WF in 2 dimensions for single-path routing is given on fig 1. At first both rates x_1 and x_2 are increased equally, until x_2 saturates its bottleneck link. Thus we can further increase only x_1 until it also hits a bottleneck link.

An extension of this scenario is introduced for example in [11] and [18]. Each flow is separately guaranteed a minimal rate. The algorithm used there for computing the max-min fair rate allocation is a modified WF. Namely, all rates are set to their minimal guaranteed values, and only the lowest rates are increased. A simple 2-dimensional example with an illustration of WF is given on fig 1.

Max-min fairness for single-rate multicast sessions is defined in [9]. This is generalized to multi-rate multicast sessions in [6]. While rates are again upper-bounded by links capacities, we are here interested in max-min fair allocation of receivers rates. A set of feasible allocations is linearly constrained, and a WF approach can be used. The geometric shape of the feasible set is essentially the same as in single-path routing.

What the above scenarios have in common is the linearity of the constraints defining the feasible set. In [19], a single-path routing scenario is considered, and each source is assigned a utility, which is an increasing and concave function of its rate. Instead of searching a max-min fair rate allocation, they look for max-min fair utility allocation. This approach is generalized in [6], where a max-min fair utility allocation is considered in the context of a multicast network. Here, the authors only required

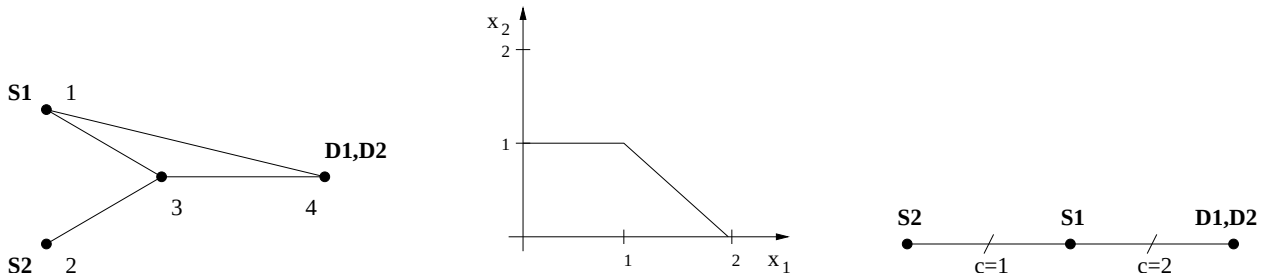

Figure 2: A simple multi-path example. Left: S1 sends to D1 over two paths, 1-3-4 and 1-4, while S2 sends to D2 over a single path 2-3-4. Center: the set of feasible rates. Right: the corresponding virtual single path problem.

that a utility function is a strictly increasing but not necessarily concave function of rate, hence the feasible set is not necessarily convex. A simple 2-dimensional example is given on fig 1. We also see that the WF algorithm can be used in this case as well.

When Bottleneck and Water-Filling Become Less Obvious. It is not always obvious how to generalize the notion of a bottleneck link and the water-filling approach to an arbitrary problem. To see why, consider a point-to-point multi-path routing scenario, where, to our knowledge, max-min fairness was not studied before. We look at the same set-up as above, but now allowing for multiple paths to be used by a single source-destination pair. The rate of communication between a source and destination is equal to the sum of rates over all used paths, and we still want a max-min fair rate allocation of source-destination rates. This is defined as above, in other words, a source-destination rate allocation is max-min fair if one cannot increase the rate of a source-destination pair without decreasing the rate of some other pair, which is already smaller. An example is given on fig 2. Let us assume that all links in the given example have capacity 1. Then, by increasing all the rates at the same pace, we will have rates of all paths equal to 1/2 when link 3-4 saturates. Now, if we continue increasing the rate over path 1-4, the rate of source destination pair 1 will be higher than the rate of source destination 2, and path 2-3-4 will loose its bottleneck.

A first question that arises is how to define a bottleneck in this case, such that the water-filling algorithm still works, if if is possible at all. Also, it is not clear if for a given definition of a bottleneck we can still claim the same as for the single-path case, i.e., that if each path has a bottleneck, then the allocation is max-min fair. Finally, we don't even know, using the existing state of the art, if a max-min fair allocation exists on an arbitrary multi-path network.

This example can be solved by observing that the max-min fair allocation depends only the set of feasible rates. Consider again the example on fig 2, left. Assume all links have unit capacity. Call $x_1 = y_1 + y_2$ the rate of source 1, and x_2 the rate of source 2, where y_1 is the rate of source 1 on path $1-4$, and y_2 on path $1-3-4$. The set of feasible rates is the set of $(x_1 \geq 0, x_2 \geq 0)$ such that there exist slack variable $y_1 \geq 0, y_2 \geq 0$ with $y_1 \leq 1, y_2 + x_2 \leq 1$ and $x_1 = y_1 + y_2$. This is an *implicit* definition, which can be made explicit by eliminating the slack variables; this gives the conditions $x_1 \leq 1, x_1 + x_2 \leq 2$ (fig 2, center). The set is convex, with a linear boundary, as on fig 1, left. We can re-interpret the original multi-path problem as a virtual single path problem (fig 2, right), and apply the existing WF algorithms. Note however that the concept of bottleneck in the virtual single path problem has lost its physical interpretation.

When Bottleneck and Water-Filling Do Not Work. Unfortunately, the trick applied when modifying the multi-path problem into a single path problem does not always work. Consider the following workload distribution example, where min-max fairness is used: a client receives data from multiple servers, and has a guaranteed minimal rate of reception. Each flow from a server to a client is constrained by link capacities. We are searching for a min-max distribution of server load. A 2-

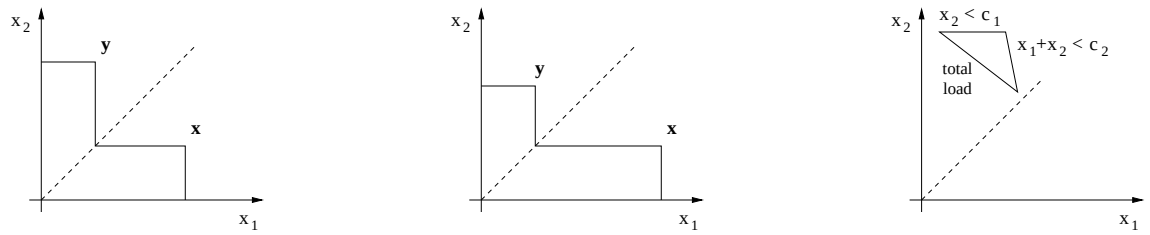

Figure 3: Feasible set examples: Left: both vectors $\vec{x}$ and $\vec{y}$ are leximin maximal and there is no max-min fair vector. Center: vector $\vec{x}$ is leximin larger than all the other vectors in the set $\mathcal{X}$, including $\vec{y}$, but still there is no max-min fair allocation. Right: example of feasible load distributions for a set of two servers.

dimensional example is given on fig 3, right. A simple visual attempt to apply WF on the set of feasible rates shows that it is not possible. A similar, but simpler example is given in [13], which focused on finding a leximax minimal allocation (we recall the definition of leximax later in the paper; we show in Section 3 that the leximax minimal allocation obtained in [13] is in fact min-max fair). Its complexity is of the order of N linear programming steps in $\mathbb{R}^N$, in the case where the feasible set is defined by linear constraints.

Next, we ask ourselves when the bottleneck argument can be made. We find that, if the set of feasible allocations has the free-disposal property (defined in Section 3), then Max-min Programming degenerates to Water Filling. Thus Free disposal corresponds to the cases where a bottleneck argument can be made, and Water Filling is the general form of all previously known centralized algorithms for such cases.

Other Examples. In Section 4.1 we consider the lifetime of nodes in a sensor network, motivated by the example introduced in [12], which studied the minimum lifetime. We study instead the max-min fair allocation of lifetimes, show that it fits in our framework, and that a max-min fair allocation does exist. It is actually an example of free-disposal implicit feasible set, thus WF can be applied after making the set explicit. In Section 4.2, we consider an example of load allocation in a peer-to-peer network; here the set does have a min-max allocation, is not free disposal, and the MP algorithm can be used.

This Paper. In Section 2 we define exactly our framework (max-min and min-max fairness in N continuous variables). We mention a number of elementary results, such as the uniqueness and the reduction of weighted max-min fairness to the unweighted case. Our derivations are based on the relation between max-min fairness and leximin ordering, a concept used in economy, which we recall. Then we consider the issue of existence. We note that there are sets on which max-min fairness does not exist, and give a large class of continuous sets on which a max-min fair allocation does exist. This class contains the compact, convex sets of $\mathbb{R}^N$, but not only. In Section 3, we give the general purpose, centralized algorithm, called Max-min Programming, and prove that if finds the max-min fair allocation in all cases where it exists. Its complexity is of the order of N linear programming steps in $\mathbb{R}^N$, in general, whenever the feasible set is defined by linear constraints. We recall the definition of Free Disposal and show that, if it holds, then Max-min Programming degenerates to the simpler algorithm of Water Filling, whose complexity is much less. Free disposal corresponds to the cases where a bottleneck argument can be made, and Water Filling is the general form of all previously known centralized algorithms for such cases. We note that WF requires the feasible set to be given in explicit form, unlike MP, and discuss the case of an implicit feasible set with disposal property. In Section 4 we give application to various networking examples, thus verifying how our framework unifies previous results, and extends the applicability of max-min fairness to new scenarios.

All our results are given for max-min fairness; they apply mutatis mutandis to min-max fairness. They are valid for weighted and unweighted max-min and min-max fairness, using the transformation

given in Section 2.1. Distributed algorithms for the computation of max-min fair allocations [7, 1] are left outside the scope of this paper. Proofs are in the full version of the paper [4], available online.

2 Max-Min and Min-Max Fairness in Euclidean Spaces

2.1 Definitions and Uniqueness

Consider a set $\mathcal{X} \subset \mathbb{R}^N$. We define the max-min and min-max fair vectors with respect to set $\mathcal{X}$ as follows:

Definition 1 *[5] A vector $\vec{x}$ is "max-min fair on set $\mathcal{X}$" if and only if*

$$(\forall \vec{y} \in \mathcal{X})(\exists s \in \{1, ..., N\})\ y_s > x_s \quad \Rightarrow \quad (\exists t \in \{1, ..., N\})\ y_t < x_t \leq x_s \tag{1}$$

i.e. increasing some component x_s must be at the expense of decreasing some already smaller component x_t.

Definition 2 *A vector $\vec{x}$ is "min-max fair on set $\mathcal{X}$" if and only if*

$$(\forall \vec{y} \in \mathcal{X})(\exists s \in \{1, ..., N\})\ y_s < x_s \quad \Rightarrow \quad (\exists t \in \{1, ..., N\})\ y_t > x_t \geq x_s \tag{2}$$

i.e. decreasing some component x_s must be at the expense of increasing some already larger component x_t.

It is clear that if $\vec{x}$ is a min-max fair vector on $\mathcal{X}$, then $-\vec{x}$ is max-min fair on $-\mathcal{X}$ and vice versa. Thus, in the remainder of the paper, we give theoretical results only for max-min fairness. Uniqueness is assured by the following proposition, the proof of which is analog to the one in [5] and is not given here.

Proposition 1 *If a max-min fair vector exists on a set $\mathcal{X}$, then it is unique.*

Weighted min-max fairness is a classical variation, defined as follows. Given some positive constants w_i (called the "weights"), a vector $\vec{x}$ is "weighted-max-min fair" on set $\mathcal{X}$, if and only if increasing one component x_s must be at the expense of decreasing some other component x_t such that $x_t/w_t \leq x_s/w_s$ [5]. This is generalized in [6], which introduces the concept of "util max-min fairness": given N increasing functions $\phi_i : \mathbb{R} \to \mathbb{R}$, interpreted as utility functions, a vector $\vec{x}$ is "util-max-min fair" on set $\mathcal{X}$ if and only if increasing one component x_s must be at the expense of decreasing some other component x_t such that $\phi_t(x_t) \leq \phi_s(x_s)$ (this is also called "weighted max-min fairness" in [15]). Consider the mapping ϕ defined by

$$(x_1, \cdots, x_N) \to (\phi_1(x_1), \cdots, \phi_N(x_N)) \tag{3}$$

It follows immediately that a vector $\vec{x}$ is util-max-min fair on set $\mathcal{X}$ if and only if $\phi(\vec{x})$ is max-min fair on the set $\phi(\mathcal{X})$, the case of weighted max-min fairness corresponding to $\phi_i(x_i) = x_i/w_i$. Thus, we now restrict our attention to unweighted max-min fairness.

2.2 Max-Min Fairness and Leximin Ordering

In the rest of our paper we will extensively use leximin ordering, a concept we borrow from economy, and which we now recall. Let us define the "order mapping" $\mathcal{T} : \mathbb{R}^N \to \mathbb{R}^N$ as the mapping that sorts $\vec{x}$ in non-decreasing order, that is: $\mathcal{T}(x_1, \cdots, x_n) = (x_{(1)}, \cdots, x_{(n)})$, with $x_{(1)} \leq x_{(2)} \cdots \leq x_{(n)}$ and for all i, $x_{(i)}$ is one of the x_js. Let us also define the lexicographic ordering of vectors in $\mathcal{X}$ by $\vec{x} \overset{lex}{>} \vec{y}$ if and only if $(\exists i)\, x_i > y_i$ and $(\forall j < i)\, x_j = y_j$. We also say that $\vec{x} \overset{lex}{\geq} \vec{y}$ if and only if $\vec{x} \overset{lex}{>} \vec{y}$ or $\vec{x} = \vec{y}$. This latter relation is a total order on $\mathbb{R}^N$.

Definition 3 *[2] Vector $\vec{x}$ is* leximin larger *than or equal to $\vec{y}$ if $\mathcal{T}(\vec{x}) \overset{lex}{\geq} \mathcal{T}(\vec{y})$.*

Definition 4 *[2] Vector $\vec{x} \in \mathcal{X}$ is* leximin maximal *on a set $\mathcal{X}$ if for all $\vec{y} \in \mathcal{X}$ we have $\mathcal{T}(\vec{x}) \overset{lex}{\geq} \mathcal{T}(\vec{y})$.*

Note that a leximin maximum is not necessarily unique. See fig 3 on the left for a counter-example.

Proposition 2 *[16] Any compact subset of $\mathbb{R}^n$ has a leximin maximal vector.*

It has been observed in [19, 11, 6] that a max-min fair allocation is also leximin maximal, for the feasible sets defined in these papers. It is generalized to an arbitrary feasible set in [16], as follows.

Proposition 3 *If a max-min fair vector exists on a set $\mathcal{X}$, then it is the unique leximin maximal vector on $\mathcal{X}$.*

Thus, the existence of a max-min fair vector implies the uniqueness of a leximin maximum. The converse is not true: see fig 3, center, for an example of a set with unique leximin maximal vector which is not max-min achievable. [16] defines a weaker version of max-min fairness, "maximal fairness"; it corresponds to the notion of leximin maximal vector, hence it is not unique, and exists on a larger class of feasible sets. We leave this weaker version outside the scope of this paper.

2.3 Existence and Max-Min Achievable Sets

As already mentioned, a number of papers showed the existence of max-min fair allocation in many cases, using different methods. We give here a generalized proof that holds on larger class of continuous sets. Note that a max-min fair vector does not exist on all feasible sets, even compact and connected. Simple counter-examples are given on fig 3, left and center. In the rest of this section we give a sufficient condition for a max-min vector to exist.

Definition 5 *A set $\mathcal{X}$ is max-min achievable if there exists a max-min fair vector on $\mathcal{X}$.*

Theorem 1 *Consider a mapping ϕ defined as in Equation 3. Assume that ϕ_i is increasing and continuous for all i. If the set $\mathcal{X}$ is convex and compact, then $\phi(\mathcal{X})$ is max-min achievable.*

Note that if $\vec{x}^*$ is max-min fair on $\mathcal{X}$ and $\vec{y}^*$ is max-min fair on $\phi(\mathcal{X})$, then in general $\vec{y}^* \neq \phi(\vec{x}^*)$.

As a special case, obtained by letting $\phi_i(x) = x$, we conclude that all convex and compact sets are max-min achievable. Taking $\phi_i(x) = x/w_i$, we also conclude that weighted max-min fairness exists on all compact, convex sets. More generally, util-max-min fairness exists on all compact, convex sets, if the utility functions are continuous (and increasing).

In [19], the utility functions ϕ_i are arbitrary continuous, increasing and concave functions. With these assumptions, the set $\phi(\mathcal{X})$ is also convex and compact. Note that in general, though, the set $\phi(\mathcal{X})$ used in the Theorem is not necessarily convex. Examples with non convex set are provided in [15] and [6].

It follows from Proposition 3 that the max-min fair vector, if it exists, is strictly Pareto optimal. It is also known that no continuous utility function exists that can represent leximin ordering [2], hence we cannot use a classical convex optimization approach to find a max-min fair vector. However, in the case of convex feasible sets, a max-min fair vector is a limiting case of utility fair allocations; this is shown in [8] for the specific case of a single path network, and in the full version of the paper [4] for the general case.

3 Max-Min Programming and Water-Filling

3.1 The Max-Min Programming (MP) Algorithm

We now present the max-min programming (MP) algorithm, which finds the max-min fair vector on any feasible set, if it exists.

1. let $S^0 = \{1, ..., N\}, \mathcal{X}^0 = \mathcal{X}$ and $n = 0$
2. **do**
3. $n = n + 1$
4. Problem MP^n: maximize T^n subject to:
$$\begin{cases} (\forall i \in S^{n-1}) & x_i \geq T^n \\ & \vec{x} \in \mathcal{X}^{n-1} \end{cases}$$
5. let $\mathcal{X}^n = \{\vec{x} \in \mathcal{X}^{n-1} \mid (\forall i \in S^{n-1})\, x_i \geq T^n, (\exists i \in S^{n-1})\, x_i > T^n\}$ and $S^n = \{i \in \{1, ..., N\} \mid (\forall \vec{x} \in \mathcal{X}^n)\, x_i > T^n\}$
6. **until** $S^n = \emptyset$
7. return the only element in $\mathcal{X}^n$

Intuitively, the algorithm maximizes in each step the minimal coordinate of the feasible vector, until all coordinates are processed. The n-th step of the algorithm is a minimization problem, called MP^n: $\mathcal{X}^n$ represents the remaining search space, and S^n represents the direction of search, in terms of coordinates that can be further increased. The algorithm always terminates if $\mathcal{X}$ is max-min achievable, and then $\mathcal{X}^n$ is reduced to one single element, which is the required max-min fair vector, as is proved in the following theorem:

Theorem 2 *The above algorithm terminates and finds the max-min fair vector on $\mathcal{X}$ if it exists, in at most N steps.*

The algorithm presented in [13] for calculating the leximax minimal allocation is a particular implementation of MP. Since the feasible set considered there is compact convex, it follows from Theorem 1 and Proposition 3 that the leximax minimal allocation obtained in [13] is in fact a min-max fair allocation.

3.2 The Water-Filling (WF) Algorithm

We now compare MP with the water-filling approach used in the traditional setting [5]. We here present a generalized version that includes minimal rate guarantees, as in [18].

We first need the concept of free-disposal. It is defined in economy as the right of each user to dispose of an arbitrary amount of owned commodities [2], or alternatively, to consume fewer resources than maximally allowed. We slightly modify it, as follows. Call $\vec{e}_i$ a unitary vector $(\vec{e}_i)_j = \delta_{ij}$.

Definition 6 *We say that a set $\mathcal{X}$ has the free disposal property if (1) there exists $\vec{m}$ with $x_i \geq m_i$ for all $\vec{x} \in \mathcal{X}$ and (2) for all $i \in \{1, ..., N\}$ and for all α such that $\vec{x} - \alpha\vec{e}_i \geq \vec{m}$, we have $\vec{x} - \alpha\vec{e}_i \in \mathcal{X}$.*

Informally, free-disposal applies to sets where each coordinate is independently lower-bounded, and requires that we can always decrease a feasible vector, as long as we remain above the lower

bounds. We now describe the Water-Filling algorithm.

0. Assume $\mathcal{X}$ is free-disposal
1. let $S^0 = \{1, ..., N\}, \mathcal{X}^0 = \mathcal{X}$ and $n = 0$
2. **do**
3. $n = n + 1$
4. Problem WF^n: maximize T^n subject to:
$$\begin{cases} (\forall i \in S^{n-1}) & x_i = \max(T^n, m_i) \\ & \vec{x} \in \mathcal{X}^{n-1} \end{cases}$$
5. let $\mathcal{X}^n = \{\vec{x} \in \mathcal{X}^{n-1} \mid (\forall i \in S^{n-1})\, x_i \geq T^n, (\exists i \in S^{n-1})\, x_i > T^n\}$
and $S^n = \{i \in \{1, ..., N\} \mid (\forall \vec{x} \in \mathcal{X}^n)\, x_i > T^n\}$
6. **until** $S^n = \emptyset$
7. return the only element in $\mathcal{X}^n$

The following theorem demonstrates the equivalence of MP and WF on free disposal sets.

Theorem 3 *Let $\mathcal{X}$ be a max-min achievable set that satisfies the free disposal property. Then, at every step n, the solutions to problems WF^n and MP^n are the same.*

Thus, under the conditions of the theorem, WF terminates and returns the same result as MP, namely the max-min fair vector if it exists. The theorem is actually stronger, since the two algorithms provide the same result at every step. However, if the free disposal property does not hold, then WF may not compute the max-min fair allocation: fig 3 (right) gives such an example.

The examples previously mentioned of single path unicast routing [5], multicast util-max-min fairness [9, 6] and minimal rate guarantee [18, 11] all have the free disposal property. Thus, the water-filling algorithm can be used, as is done in all the mentioned references. In contrast, the load distribution example [13] is not free-disposal, and all we can do is use MP, as is done in [13] in a specific example.

The multi-path routing example also has the free disposal property, but the feasible set is defined implicitly. We discuss the implications of this in the next section.

3.3 Complexity Of The Algorithms In Case Of Linear Constraints

Let us assume first that $\mathcal{X}$ is an n-dimensional feasible set defined by m linear inequalities. Each of the n steps of MP algorithm is a linear programming problem, hence the overall complexity is $O(nLP(m))$, where $LP(m)$ is the complexity of linear programming. The WF algorithm also has n steps, each of complexity $O(m)$ hence the complexity of WF is $O(nm)$. Linear programming has solutions of exponential complexity in the worst case, however in most practical cases there are solutions with polynomial complexity.

Assume second that $\mathcal{X}$ is defined implicitly, with an l-dimensional slack variable (for an example scenario see multi path case on figure 2). We can use MP, which works on implicit sets, and with complexity $O(nLP(m))$. If the set is free-disposal, we can also use WF, but we need to find an explicit characterization of the feasible set. This itself is a linear programming problem. Once having an explicit characterization, the remaining complexity of WF is still $O(mn)$. In practice, we are likely to be interested in finding the values of slack variables at the max-min fair vector; this, which is again of complexity $LP(m)$. However, it is sufficient to make the set explicit only once for a given problem. We conclude that in many practical problems, it is likely to be faster to make the set of constraints explicit and use WF rather than MP.

4 Example Scenarios

In this section we provide examples arising in a networking context, which were not previously studied, and to which our theory applies.

4.1 Minimum Energy Wireless Network

Our first novel example is the lifetime of a sensor network. Assume that we have a wireless network represented with a graph, with a set of communicating source-destination pairs. Let us denote by e_l the energy needed to transfer one bit over wireless link l, by E_n the initial energy of node n, by Y_p the total amount of bits transferred over path p, and by X_s the total amount of bits transferred from a source to a destination. Regarding the lifetime of such a network, a natural goal would be to maximize the amount of information transferred from a source to a destination. A feasible information allocation is thus

$$\mathcal{X} = \{X_s \,:\, E_n \geq \sum_l a_{n,l} e_l \sum_p b_{l,p} Y_p, X_s = \sum_p d_{s,p} Y_p\},$$

and we search for a max-min fair vector on set $\mathcal{X}$. Set $\mathcal{X}$ is convex so it is also max-min achievable. It also has a free disposal property so max-min fair allocation can be found by either WF or MP since $\mathcal{X}$ is not given explicitly (see section 3.3). Bottlenecks of the explicit feasible set in this case are minimal cuts with respect to the available energy of nodes, instead of capacities of links, as in the multi-path case.

This model is an extension of the model presented in [12]. In [12] they maximize only the shortest lifetime, hence solve only MP^1. This is not useful in case of large networks where a death of one node cannot be treated as a death of the network itself. Also, in [12] they optimize lifetime given the traffic matrix, whereas we allow an arbitrary traffic matrix and we optimize the amount of information transferred. Thus, [12] is a special case of our model presented here.

4.2 Load Distribution In P2P Systems

Let us consider a peer-to-peer network or a context-distribution system, where several servers supply a single user with parts of a single data stream. There is a minimal rate a user needs to achieve, and there is an upper bound on each flow given by a network topology and link capacities. We are interested in equalizing the load of servers. Using a similar notation as in rate allocation examples, we can represent the feasible set of load allocation as

$$\mathcal{X} = \{\vec{x} \,:\, (\exists \vec{y}, \vec{z})\ A\vec{y} \leq \vec{c}, B\vec{y} = \vec{z}, \vec{z} \geq \vec{m}\}, \quad A, B \geq 0, \tag{4}$$

where $\vec{x}$ is the total load on a server, $\vec{y}$ is the flow from a server to a client, and $\vec{z}$ is the total traffic received by a server. We are interested in the min-max fair vector on set $\mathcal{X}$. This set is convex, hence it is the min-max achievable. It does not have a free disposal property in general (see fig 3 on the right for an example), hence WF is not applicable. Min-max fair allocation can be found my means of the MP algorithm.

Note that this form of feasible set is unique in that it introduces both upper and lower bounds on a sum of components of $\vec{x}$ and as such is more general than the feasible sets in the above presented examples, such as [13].

5 Conclusion

We have given a general framework that unifies several results on max-min and min-max fairness encountered in networking examples. We have extended the framework to account for new examples arising in mobile and peer-to-peer scenarios. We have elucidated the role of bottleneck arguments, and made the relation to the free-disposal property; we have shown that the bottleneck argument is not essential to the definition of max-min fairness, contrary to popular belief. However, when it holds, it allows to use simpler algorithms. We have given a general purpose algorithm (MP) for computing the max-min fair vector whenever it exists, and showed that it degenerates to the classical water-filling algorithm, when free disposal holds. The existence of a max-min fair vector is not always guaranteed, even on compact sets. We have found a class of compact sets on which max-min fairness does exist; it remains to extend the class to other useful cases, such as discrete sets [16]. Finally, we have focused on centralized algorithms for calculating max-min and min-max fair allocations. It will be interesting to explore their distributed counterparts.

References

[1] A. Charny. An algorithm for rate allocation in a packet-switched network with feedback. *M.S. thesis*, MIT, May 1994.

[2] A. Mas-Colell, M. Whinston, J. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[3] ATM Forum Technical Committee. "Traffic Management Specification - Version 4.0". *ATM Forum/95-0013R13*, February 1996.

[4] B. Radunović and J.-Y. Le Boudec. "A Unified Framework for Max-Min and Min-Max Fairness with Applications". *Technical report IC-200248*, EPFL, July 2002. http://lcawww.epfl.ch/Publications/Radunovic/TR02_048.ps

[5] D. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, 1987.

[6] D. Rubenstein, J. Kurose, D. Towsley. "The Impact of Multicast Layering on Network Fairness". *IEEE/ACM Transactions on Networking*, 10(2):169–182, Apr. 2002.

[7] E. Hahne. "Round-Robin Scheduling for Max-Min Fairness in Data Networks". *IEEE Journal on Selected Areas in Communications*, 9(7):1024–1039, Sept. 1991.

[8] F. P. Kelly, A.K. Maulloo and D.K.H. Tan. "Rate control in communication networks: shadow prices, proportional fairness and stability". *Journal of the Operational Research Society*, 49:237–252, 1998.

[9] H. Tzeng and K. Siu. "On Max-Min Fair Congestion Control for Multicast ABR Service in ATM". *IEEE Journal on Selected Areas in Communications*, 15(3):545–556, April 1997.

[10] J. Mo, J. Walrand. "Fair end-to-end window-based congestion control". *IEEE/ACM Transactions on Networking*, 8(5):556–567, Oct. 2000.

[11] J. Ros and W. Tsai. "A Theory of Convergence Order of Maxmin Rate Allocation and an Optimal Protocol". In *INFOCOM'01*, pages 717–726, 2001.

[12] J.H. Chang and L. Tassiulas. "Energy Conserving Routing in Wireless Ad-hoc Networks". In *INFOCOM'00*, pages 22–31, 2000.

[13] L. Georgadis, et al. "Lexicographically Optimal Balanced Networks". In *INFOCOM'01*, pages 689–698, 2001.

[14] L. Tassiulas and S. Sarkar. "Maxmin Fair Scheduling in Wireless Networks". In *INFOCOM'02*, 2002.

[15] P. Marbach. "Priority Service and Max-Min Fairness". In *INFOCOM'02*, 2002.

[16] S. Sarkar and L. Tassiulas. "Fair Allocation of Discrete Bandwidth Layers in Multicast Networks". In *INFOCOM'00*, pages 1491–1500, 2000.

[17] Xiao Long Huang, Brahim Bensaou. "On Max-min Fairness and Scheduling in Wireless Ad-Hoc Networks: Analytical Framework and Implementation". In *Proceedings MobiHoc'01*, Long Beach, California, October 2001.

[18] Y. Hou, H. Tzeng, S. Panwar. "A Generalized Max-Min Rate Allocation Policy and Its Distributed Implementation Using the ABR Flow Control Mechanism". In *INFOCOM'98*, pages 1366–1375, 1998.

[19] Z. Cao and E. Zegura. "Utility Max-Min: An Application-Oriented Bandwidth Allocation Scheme". In *INFOCOM'99*, pages 793–801, 1999.

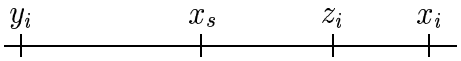

Figure 4: Choice of constants

A Proofs

Proof of theorem 1: Let $\vec{x} \in \phi(\mathcal{X})$ be a vector such that for all $\vec{y} \in \phi(\mathcal{X})$ we have $\mathcal{T}(\vec{x}) \overset{lex}{\geq} \mathcal{T}(\vec{y})$. Such vector exists according to proposition 2, since set $\mathcal{X}$ is compact. In order to prove the theorem, we proceed by contradiction, assuming that there exist $\vec{y}$ and an index $s \in \{1, ..., N\}$ such that $y_s > x_s$ and for all $t \in \{1, ..., N\}$, $x_t \leq x_s$ we have $y_t \geq x_t$. We then define a permutation $\pi : \{1, ..., N\} \rightarrow \{1, ..., N\}$ such that for all $i < j$, $\vec{x}_{\pi(i)} \leq \vec{x}_{\pi(j)}$, and either $x_{\pi(l)} < x_{\pi(l+1)}$ or $l = N$, where $l = \pi^{-1}(s)$.

Next, let us define vector $\vec{z}(\alpha) = \phi(\alpha\phi^{-1}(\vec{x}) + (1-\alpha)\phi^{-1}(\vec{y}))$. It is easy to verify that for $\alpha \in (0, 1)$, $\vec{z}(\alpha)$ belongs to $\phi(\mathcal{X})$ and for all $i \in \{1, ..., N\}$, $\min(x_i, y_i) < \vec{z}(\alpha)_i < \max(x_i, y_i)$, due to convexity of $\mathcal{X}$ and strictly increasing properties of functions ϕ_i. Also, for all i let us pick an arbitrary α_i satisfying

$$\alpha_i \in \begin{cases} \left(\frac{\phi_i^{-1}(x_s)-\phi_i^{-1}(y_i)}{\phi_i^{-1}(x_i)-\phi_i^{-1}(y_i)}, 1\right), & x_s \in [y_i, x_i), \\ [0, 1), & \text{otherwise} \end{cases}$$

and we call $\alpha_m = \max_i(\alpha_i)$ and $\vec{z} = \vec{z}(\alpha_m) \in \phi(\mathcal{X})$ (since $\alpha_m \in [0, 1)$). A choice of constant is depicted on a fig 4.

We now have the following properties

$$\begin{array}{rcll} z_{\pi(i)} & \geq & x_{\pi(i)}, & \text{for } i < l, \\ z_{\pi(i)} & > & x_{\pi(l)}, & \text{for } i \geq l. \end{array}$$

We first notice that for all i, $z_{\pi(i)} \geq x_{\pi(1)}$, and as $\mathcal{T}(\vec{x}) \overset{lex}{\geq} \mathcal{T}(\vec{z})$ we conclude that $z_{(1)} = z_{\pi(1)} = x_{\pi(1)}$. Next, assuming that for some $i < l$ and for all $j < i$ we have $z_{(j)} = z_{\pi(j)} = x_{\pi(j)}$, then again as for all $j \geq i$, $z_{\pi(j)} \geq x_{\pi(i)}$, and $\mathcal{T}(\vec{x}) \overset{lex}{\geq} \mathcal{T}(\vec{z})$ we conclude that $z_{(i)} = z_{\pi(i)} = x_{\pi(i)}$. Hence, by induction we have proved that for all $i < l$ we have $z_{(i)} = z_{\pi(i)} = x_{\pi(i)}$. Finally, since for all $i \geq l$ we have $z_{\pi(i)} > x_{\pi(l)}$, hence $z_{(i)} > x_{\pi(l)}$ we necessarily have that $\mathcal{T}(\vec{z}) \overset{lex}{>} \mathcal{T}(\vec{x})$, which brings us to the contradiction.

Therefore, we conclude that a leximin maximal vector on a set $\mathcal{X}$ is also a max-min fair vector, and set $\mathcal{X}$ is max-min achievable.

q.e.d.

Lemma 1 *For all n, $\vec{x}, \vec{y} \in \mathcal{X}^n$ and $t \in \{1, ..., N\}$ such that $x_t \leq T^n$, we have $y_t \geq x_t$.*

Proof: We prove lemma by induction over n. If $n = 1$, we have for all t, $x_t \geq T^1$ and $y_t \geq T^1$, hence for $x_t = T^1$, we have $y_t \geq x_t$. Next assume the above is true for $n-1$. We know that for all t such that $x_t < T^n$ we also have $x_t \leq T^{n-1}$, hence by assumption we have $y_t \geq x_t$. Finally, if for some t, $x_t = T^n$ then $y_t \geq T^n$ or else we have a contradiction with the definition of T^n.

q.e.d.

Lemma 2 *If $\vec{x}$ is max-min fair vector on $\mathcal{X}$ then for all n such that $\mathcal{X}^n \neq \emptyset$, $\vec{x} \in \mathcal{X}^n$.*

Proof: We prove lemma by induction. If $\vec{x} \notin \mathcal{X}^1$ then $\vec{x}$ is not leximin maximal, hence the contradiction. Let us next assume $\vec{x} \in \mathcal{X}^{n-1}$ and $\vec{x} \notin \mathcal{X}^n$, where $\mathcal{X}^n \neq \emptyset$. Then there exists $\vec{y} \in \mathcal{X}^n$ and $s \in S^n$ such that $y_s > x_s$. Also, by lemma 1, for all $t \in \{1, ..., N\}$ such that $x_t \leq T^n$, we have $y_t \geq x_t$. This contradicts the assumption that $\vec{x}$ is max-min fair which proves the lemma.

q.e.d.

Proof of theorem 2: If set $\mathcal{X}$ is max-min achievable, it is then also upper-bounded hence there exists a solution to each MP^i. Let us call $\vec{x}$ max-min fair vector on $\mathcal{X}$ and denote $S_y^n = \{i \in \{1, ..., N\} : y_i > T_i^n\}$.

We know by construction of sequence $\{S^n\}$ that $S^n \subseteq S^{n-1}$. We next prove that if max-min fair vector $\vec{x}$ exists, then for all n such that $S^n \neq \emptyset$ we have $S^{n-1} \neq S^n$, and let us assume the contrary, $S^{n-1} = S^n$. We then have the following:

1. $\mathcal{X}^{n-1} = \mathcal{X}^n$.
2. There exists $\vec{y} \in \mathcal{X}^n$ such that $S_y^n \neq S_x^n \cap S_y^n$.
3. There exists $s \in \{1, ..., N\}$ such that $y_s > x_s = T^n$.

Assume $\mathcal{X}^n \subset \mathcal{X}^{n-1}$, it implies $T^{n+1} > T^n$, which contradicts with the fact that T^n solves MP^n, hence we prove (1). Next, assume that $S_x^n = S^n \neq \emptyset$, hence for all $s \in S^n$, $x_s > T^n$. This also contradicts with the fact that T^n solves MP^n, hence there exists $\vec{y} \in \mathcal{X}^n$ such that $S_y^n \neq S_x^n \cap S_y^n$. Also, for all $s \in S_y^n \setminus S_x^n \neq \emptyset$ we have $y_s > x_s = T^n$ hence (3). From (3) and lemma 1 we see that $\vec{x}$ is not max-min fair which contradicts with $S^{n-1} = S^n$.

We conclude that sequence $|S^n|$ decreases and we will have $S^n = \emptyset$ in at most N steps. We also notice that for every $i \in \{1, ..., N\}$ there exists m such that $i \in S^{m-1}$ and $i \notin S^m$. From $i \in S^{m-1}$ we have that for all $\vec{x} \in \mathcal{X}^m$, $x_i \leq T^m$. From $i \in S^m$ we have that for all $x \in \mathcal{X}^{m-1}$ we have $x_i \geq T^m$ in the constraints for MP^m. Now as for all n, $\mathcal{X}^n \subseteq \mathcal{X}^{n-1}$ we have that for all $n \geq m$ and $\vec{x} \in \mathcal{X}^n$ we have $x_i = T^m$. Once we have $S^n = \emptyset$ it means that all components of vectors in $\mathcal{X}^n$ are fixed hence $|\mathcal{X}^n| = 1$. According to lemma 2, this single vector in $\mathcal{X}^n$ is also max-min fair on $\mathcal{X}$.

q.e.d.

Proof of theorem 3: Let us call T_{MP}^1 the solution to the MP^1 and T_{WF}^1 the solution to the WF^1. T_{WF}^1 is obviously achievable in MP^1 so we have $T_{MP}^1 \geq T_{WF}^1$. Suppose that $T_{MP}^1 > T_{WF}^1$. This implies that for all $s \in \{1, ..., N\}$ we have $(\vec{x}_{MP}^1)_s \geq T_{MP}^1$. Due to free disposal property, we can successively decrease each of components of $\vec{x}$ larger than corresponding lower bound in $\vec{m}$, until arriving to a vector $\vec{y}$, $y_i = \max(T_{MP}^1, m_i)$. This vector is feasible which contradicts the optimality of T_{WF}^1. The same reasoning can be applied to the successive algorithm steps, by decreasing the dimension of the feasible set.

q.e.d.

B Max-min fairness as a limiting case of utility fairness

Finally, we show that max-min fair vector is a limiting case of utility fair vector, thus generalizing a result from [10].

Let f_m be a family of increasing, concave and differentiable functions defined on $\mathbb{R}^+$. Assume that, for any fixed numbers x and δ,

$$\lim_{m \to +\infty} \frac{f_m(x)}{f_m(x+\delta)} = 0, \tag{5}$$

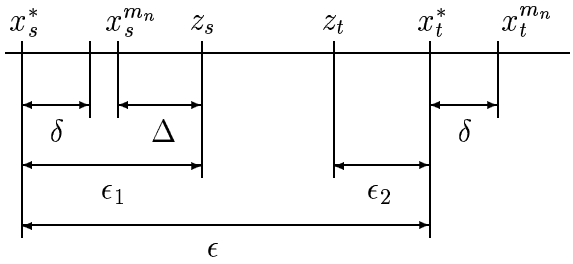

Figure 5: Choice of constants

$$\lim_{m \to +\infty} \frac{f'_m(x)}{f'_m(x+\delta)} \quad = \quad 0. \tag{6}$$

The assumptions (5) and (6) are satisfied if for example f_m is defined by $f_m(x) = c - g(x)^m$ where c is a constant and g is differentiable, decreasing, convex and positive function (e.g. $g(x) = 1/x$).

Let us consider a convex set $\mathcal{X}$ and let $\vec{x}^m$ be the utility-fair vector on $\mathcal{X}$, that is the unique vector which maximizes $\sum_{i=1}^N f_m(x_i)$ The following theorem shows that a max-min fair vector is then a limiting case of utility fair vector for the set of utilities we constructed above:

Theorem 4 *The set of utility-fair vectors $\vec{x}^m$ converges toward the max-min fair vector as m tends to $+\infty$.*

Proof: We first prove that if $\vec{x}^*$ is an accumulation point for the set of vectors $\vec{x}^m$, then $\vec{x}^*$ is the max-min fair vector, and to prove that we assume the contrary, that exists a vector $\vec{y}$ and a corresponding index $s = s(y)$ such that $y_s > x_s$ and for all $t \in S$ $x_t \leq x_s \Rightarrow x_t \leq y_t$. Let us call $T_s(x) = \{t \in \{1, ..., N\} \,|\, x_t > x_s\}$. If set $T_s(x)$ is empty, we pick an arbitrary $\epsilon > 0$, else we call $t = \arg\min_{t \in T_s(x)} x_t^*$ and $\epsilon = x_t^* - x_s^*$.

Next, similarly as in proof of theorem 1, we pick an arbitrary $\alpha < \epsilon/(y_s - x_s^* + x_t^* - y_t)$, and we call $\vec{z} = (1-\alpha)\vec{x} + \alpha\vec{y}$. If $\epsilon_1 = z_s - x_s^*$ and $\epsilon_2 = x_t^* - z_t$, it is easy to verify that $\epsilon > \epsilon_1 + \epsilon_2$. Also, $\vec{z} \in \mathcal{X}$ due to convexity of $\mathcal{X}$.

Finally we pick an arbitrary $\delta > 0$, $\Delta > 0$ such that $\delta + \Delta < \epsilon_1$, and some n_0 such that for all $n \geq n_0$ and for all $u \in \{1, ..., N\}$ we have $|f_{m_n}(x_u^*) - f_{m_n}(x_u^{m_n})| < \delta$. The choice of the above constants is depicted on fig 5.

Now we show that for n large enough we have a contradiction with optimality of $\vec{x}^{m_n}$. Consider the expression A defined by

$$A = \sum_{t \in \{1,...,N\}} \left(f_{m_n}(z_t) - f_{m_n}(x_t^{m_n}) \right).$$

From the optimality of $\vec{x}^{m_n}$ we have $A \leq 0$. Also

$$A \quad \geq \quad f_{m_n}(z_s) - f_{m_n}(x_s^{m_n}) + \sum_{t \in T_s(x)} \left(f_{m_n}(z_t) - f_{m_n}(x_t^{m_n}) \right). \tag{7}$$

From the theorem of intermediate values, there exist numbers c_s^n and c_t^n for all $t \in T_s(x)$ such that

$$\begin{aligned} & x_s^{m_n} \leq c_s^n \leq z_s, \\ & f_{m_n}(z_s) - f_{m_n}(x_s^{m_n}) = f'_{m_n}(c_s^n)(z_s - x_s^{m_n}). \end{aligned}$$

and

$$
\begin{aligned}
& z_t \leq c_t^n \leq x_t^{m_n}, \\
& f_{m_n}(x_t^{m_n}) - f_{m_n}(z_t) = f'_{m_n}(c_t^n)(x_t^{m_n} - z_t).
\end{aligned}
$$

where f'_{m_n} is the right derivative of f_{m_n}. From the above we have

$$
\begin{aligned}
f_{m_n}(z_s) - f_{m_n}(x_s^{m_n}) &> \Delta f'_{m_n}(z_s), \\
f_{m_n}(x_t^{m_n}) - f_{m_n}(z_t) &< (\delta + \epsilon_2) f'_{m_n}(z_t).
\end{aligned}
$$

thus from eq. (7)

$$
\begin{aligned}
A &> \Delta f'_{m_n}(z_s) - (\delta + \epsilon_2) M f'_{m_n}(z_t) \\
&> f'_{m_n}(z_s) \left(\Delta - (\delta + \epsilon_2) M \frac{f'_{m_n}(z_t)}{f'_{m_n}(z_s)} \right)
\end{aligned}
$$

where M is the cardinality of set $T_s(x)$. Now from eq. (5), the last term in the above equation tends to Δ as n tends to infinity. Now $f'_{m_n} > 0$ from our assumption thus, for n large enough, we have $A > 0$, which is the required contradiction.

The set of vectors $\vec{x}^m$ is in a compact set $\mathcal{X}$ hence it has at least once accumulation point. Since each accumulation point is also max-min fair, and from property 1 we have the uniqueness of a max-min fair vector we conclude that the set of vectors $\vec{x}^m$ has a unique accumulation point hence

$$
\lim_{m \to +\infty} \vec{x}^m = \vec{x}^*.
$$

q.e.d.