Fair and Efficient Allocations with Limited Demands

Sushirdeep Narayana, Ian A. Kash
Department of Computer Science, University of Illinois at Chicago, USA
snaray25@uic.edu, iankash@uic.edu

Abstract

We study the fair division problem of allocating multiple resources among a set of agents with Leontief preferences that are each required to complete a finite amount of work, which we term "limited demands." We examine the behavior of the classic Dominant Resource Fairness (DRF) mechanism in this setting and show it is fair but only weakly Pareto optimal and inefficient in many natural examples. We propose as an alternative the Least Cost Product (LCP) mechanism, a natural adaptation of Maximum Nash Welfare to this setting. We characterize the structure of allocations of the LCP mechanism in this setting, show that it is Pareto efficient, and that it satisfies the relatively weak fairness property of sharing incentives. While we prove it satisfies the stronger fairness property of (expected) envy freeness in some special cases, we provide a counterexample showing it does not do so in general, a striking contrast to the "unreasonable fairness" of Maximum Nash Welfare in other settings. Simulations suggest, however, that these violations of envy freeness are rare in randomly generated examples.

Introduction

Large computational tasks, including those found in cloud computing systems, data centers, and high-performance computing clusters, require multiple heterogeneous resources such as CPU, memory, and network bandwidth. For these systems to be effective, it is important that these and other resources be allocated among various tasks in a fair and efficient manner.

One popular approach to resource allocation in this setting is the Dominant Resource Fairness (DRF) mechanism, which achieves a number of desirable properties including efficiency (Pareto optimality), fairness (sharing incentives and envy-freeness), and strategy-proofness (Ghodsi et al. 2011; Parkes, Procaccia, and Shah 2015). However, these results are obtained under the assumption that the utility of each task is determined by the amount of resources it receives. We argue that a more natural model for many applications is that they have limited demands: each task has a finite amount of work to do and would like to complete as soon as possible. Such a model can be applied to the frameworks of job schedulers of modern computing clusters such as those used in Hindman et al. (2011), Vavilapalli et al. (2013), and Grandl et al. (2016).

The addition of limited demands does not change the fairness and incentive properties, but has a substantial effect on efficiency. Consider the following simple example. There are two agents and one (divisible) resource, and each agent needs a single unit of the resource to complete its task. The DRF allocation evenly shares the resource between the two agents, so both complete at time 2. Instead, a shortest job first approach will have one agent complete at time 1 and the other complete at time 2. This makes one agent strictly better off and the other no worse off showing that the result of DRF is no longer Pareto optimal in this setting. Of course, this allocation is not envy-free, as the agent scheduled second envies the agent scheduled first. Nevertheless, if we randomize which agent is scheduled first, it would be envy-free in expectation.

While shortest job first is envy-free in expectation and Pareto efficient, with multiple resources it too can pass up opportunities to make desirable trade-offs. Consider two tasks of similar length: one memory-limited and one CPU-limited. Shortest job first will pass up the opportunity to run both at the same time, which would slightly slow down the first completion time but substantially speed up the second. By most metrics of interest (e.g. mean completion time or makespan) this is an improvement.

Thus, a natural goal is an algorithm that can provide a trade off between sharing and shortest job first to achieve efficiency as well as fairness. Prior systems work (Grandl et al. (2016)) has developed heuristic solutions to this problem; our contribution is a principled fair division approach. We propose the Least Cost Product (LCP) mechanism, which minimizes the product of completion times (costs) in the same way Maximum Nash Welfare (MNW) mechanisms maximize the product of utilities.[1] We provide a characterization of the structure of solutions it finds and show that the LCP mechanism satisfies sharing incentives and is Pareto optimal but not strategy-proof. We show that it satisfies envy-freeness in expectation for two special cases: when there is only a single resource or when there are only two agents. When more agents or resources are involved, we

[1]In fact, if utility is defined to be the reciprocal of completion time, our mechanism chooses exactly the MNW solution.

present an example of an LCP allocation with envy. This is a striking contrast to the fairness properties satisfied by the Maximum Nash Welfare allocations in other settings. All adaptations of Maximum Nash Welfare we are aware of in divisible settings, such as Competitive Equilibrium with Equal Incomes (CEEI) for dividing divisible goods with linear additive utilities (Varian 1974; Moulin 2004) and Leontief utilities (Ghodsi et al. 2011; Goel, Hulett, and Plaut 2019); the competitive rule for dividing bads with linear additive disutilities (Bogomolnaia et al. 2019); and competitive allocations for dividing mixed manna with concave, continuous and 1-homothetic (dis)utilities (Bogomolnaia et al. 2017) satisfy strong envy-freeness guarantees. We believe our setting of "limited demands" is the first natural divisible setting where an adaptation of Maximum Nash Welfare has resulted in allocations not being envy-free. However, we show in simulations on randomly generated instances that the LCP allocations that violate envy-freeness are rare.

Related Work

Ghodsi et al. (2011) introduced the Dominant Resource Fairness (DRF) mechanism for allocation of resources when the agents have Leontief preferences. Parkes, Procaccia, and Shah (2015) generalized the DRF mechanism to more expressive settings and established stronger fairness properties like Group Strategy-Proofness. Kash, Procaccia, and Shah (2014) extended the DRF mechanism for dynamic settings. These works do not consider any notion of a limited amount of resource required by an agent to complete its work.

Varian (1974) explored the CEEI mechanism which is a market-based interpretation to allocate goods and showed that the allocations are envy-free and Pareto efficient. Arrow and Intriligator (2000) found that the CEEI allocations coincide with allocations where the objective is to maximize the Nash welfare. Caragiannis et al. (2016) showed that for allocating indivisible items the maximum Nash welfare (MNW) solution achieves both EF1 (envy-freeness up-to one good) and Pareto optimality simultaneously, motivating them to call it *unreasonably fair*. For a survey, see Moulin (2019). Conitzer, Freeman, and Shah (2017) worked on the fair division problem for public decision making and showed that the MNW solution approximates or satisfies various notions of proportionality. Conitzer et al. (2019) introduced the notion of *group fairness* which generalizes envy freeness to groups of agents. They showed that locally optimal Nash welfare allocations satisfy "up-to one good" style relaxations of *group fairness*. These works illustrate the promising nature of maximizing Nash welfare. However, none of these works observe the failures of fairness we do.

More closely related to our work, Friedman and Henderson (2003) analyzed the problem of single resource allocation when agents with jobs that required different service times arrived in an online manner. They proposed the Fair Sojourn Protocol which is more efficient and fair compared to the Shortest Remaining Processing Time and Processor Sharing protocols (corresponding to Shortest Job First and DRF in our setting respectively). We explore a generalized, but offline version of their problem where agents have demands on multiple heterogeneous resources.

Grandl et al. (2016) proposed an *altruistic* two-level scheduler called Carbyne that schedules jobs demanding multiple resources where each job takes the form of a Directed Acyclic Graph (DAG) with vertices representing tasks, and edges denoting dependencies. They show that experimentally Carbyne performs better than the DRF and the Shortest Job First (SJF) when evaluating inter-job fairness, performance and cluster efficiency. This work inspired our examination of limited demands.

Finally, Bogomolnaia et al. (2017) generalized the competitive division mechanism for the fair division of mixed manna where items are goods to some agents, but bads or satiated to others. They prove that when the items consist of only bad utility profiles, the competitive profiles are the critical points of the product of disutilities on the efficiency frontier. Our characterization of the LCP mechanism, which is also an allocation of bads, has a similar flavor.

Preliminaries

Let $\mathcal{N} = \{1, 2, 3, ..., n\}$ denote the set of agents and $\mathcal{R} = \{1, 2, ..., m\}$ denote the set of heterogeneous resources available in the system. The resources are assumed to be divisible. Each resource in the set $\mathcal{R}$ has a unit capacity. An agent in the system demands various resources be allocated to it in fixed proportions, known as Leontief preferences. Let $\mathbf{D}$ represent the demands of all the agents with each element d_{ir} denoting the fraction of resource r required by agent i. We assume these are normalized so that the demand vector for agent i denoted by $\mathbf{d_i} = \langle d_{i1}, d_{i2}, ..., d_{im} \rangle$ has $d_{ir} = 1$ for some r. The number of tasks that are required to be completed by an agent i is k_i and is represented by an $n \times n$ diagonal matrix $\mathbf{K}$ with the i-th diagonal element equal to k_i.[2] The amount of resources that needs to be consumed by an agent i in order to complete its job is given by $\mathbf{w_i} = k_i \mathbf{d_i}$. In matrix notation, this becomes $\mathbf{W} = \mathbf{KD}$.

Definition 1. An instantaneous allocation $\mathbf{A} \subseteq \mathbb{R}^{n\times m}$ allocates a portion A_{ir} of resource type r to agent i subject to the feasibility conditions, $\sum_{i\in\mathcal{N}} A_{ir} \leq 1, \ \forall r \in \mathcal{R}$; and $A_i = \lambda_i \mathbf{d_i}, \ \forall i \in \mathcal{N}$ and for some $\lambda_i \in \mathbb{R}_{\geq 0}$.

Definition 2. A resource allocation mechanism is a function f that takes as inputs the amount of resources $\mathbf{W}$ that are required by the agents and maps them to allocations as outputs, that is, $f : \mathbb{R}^{n\times m} \to (\mathbb{R}^{n\times m})^{\mathbb{R}_{\geq 0}}$ such that $f_i(\mathbf{W}) = \mathbf{A}_i(\cdot)$, where $\mathbf{A}_i(t)$ denotes the allocation of agent i at time t, which we refer to as its *instantaneous allocation*.

This definition assumes that resources are fully divisible, which is not necessarily true in real systems. In practice this can be handled by keeping allocations as close as possible to the divisible ideal over time (Hindman et al. 2011).

The cost of an agent characterizes the loss suffered by that agent and is equal to the completion time of the agent for its resource allocation. The cost $c_i(f_i(\mathbf{W})) = t_i$ for an agent i under the allocation mechanism f can be calculated as the value of t_i that solves $\mathbf{w_i} = \int_0^{t_i} \mathbf{A}_i(t)dt$.

[2]The normalization of d_i means that k_i need not be an integer.

In order to evaluate and compare the allocations produced by various resource allocation mechanisms, we define four standard fairness and efficiency properties for our setting:

- Pareto Optimality (PO): An allocation $\mathbf{A}(\cdot)$ is said to be Pareto optimal if there is no alternative allocation $\mathbf{A}^{'}(\cdot)$ which can make at least one agent strictly better off without making any other agent worse off. Formally,
  $\forall \mathbf{A}^{'}(\cdot), \quad (\exists i \in \mathcal{N}, \quad c_i(\mathbf{A}_i^{'}(\cdot)) < c_i(\mathbf{A}_i(\cdot)))$
  $\implies (\exists j \in \mathcal{N}, \quad c_j(\mathbf{A}_j^{'}(\cdot)) > c_j(\mathbf{A}_j(\cdot)))$.
  An allocation $\mathbf{A}(\cdot)$ is weakly Pareto optimal if there is no alternative allocation $\mathbf{A}^{'}(\cdot)$ which would strictly benefit all the agents. That is,
  $\nexists \mathbf{A}^{'}(\cdot)$, where $\forall i \in \mathcal{N}, \quad c_i(\mathbf{A}_i^{'}(\cdot)) \; < \; c_i(\mathbf{A}_i(\cdot))$
- Sharing Incentives (SI): By abuse of notation, we will refer to $\mathbf{A}_{\text{SI}}(\cdot) = \mathbf{A}_{\text{SI}} = \langle \frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n} \rangle$ as the instantaneous allocation that splits all the *m* resources equally among all the agents. An allocation mechanism satisfies Sharing Incentives if, $\quad c_i(\mathbf{A}_i(\cdot)) \; \leq \; c_i(\mathbf{A}_{\text{SI}}) = n \cdot k_i, \quad \forall i \in \mathcal{N}$. In other words, if the cost for each agent's allocation under the allocation mechanism is at-most the cost encountered when there is an equal split of the resources among agents, the allocation mechanism satisfies SI.
- Envy-Freeness (EF): An allocation mechanism is envy-free (EF) if $\quad \forall i,j \in \mathcal{N}, \quad c_i(\mathbf{A}_i(\cdot)) \quad \leq \quad c_i(\mathbf{A}_j(\cdot))$. That is, each agent would never strictly prefer the resource allocation received by another agent.

  A randomized allocation mechanism is envy-free in expectation if, $\forall i,j \in \mathcal{N}, \quad \mathbb{E}[c_i(\mathbf{A}_i(\cdot))] \; \leq \; \mathbb{E}[c_i(\mathbf{A}_j(\cdot))]$, where the expectation is taken over the randomness of the mechanism. In particular, when there are multiple agents with the same amount of work, the allocation mechanism may have multiple optimal solutions. In these cases, as in the example in the introduction, each individual choice has envy but a uniform random selection from them will be envy-free in expectation.
- Strategy-Proofness (SP): An allocation mechanism is strategy-proof (SP) if an agent can never benefit from reporting a false demand, regardless of the demands of the other agents. Given the demands $\mathbf{D}$ and the number of tasks $\mathbf{K}$, let $f(\mathbf{KD})$ be the resulting allocation due to the mechanism *f*. Suppose an agent $i \in \mathcal{N}$ changes to an untruthful demand $\mathbf{d}_i^{'}$ while the demands of the other agents and the number of tasks $\mathbf{K}$ stay the same. The new demand matrix is denoted by $\mathbf{D}^{'}$. The allocation mechanism *f* is said to be SP if,
  $\forall i \in \mathcal{N}, \, \forall d_i, \; c_i(f_i(\mathbf{KD})) \leq c_i(f_i(\mathbf{KD}^{'}))$,
  where $\mathbf{D} = \begin{bmatrix} \mathbf{d}_1 \\ ... \\ \mathbf{d}_i \\ ... \\ \mathbf{d}_n \end{bmatrix}$, and $\mathbf{D}^{'} = \begin{bmatrix} \mathbf{d}_1 \\ .. \\ \mathbf{d}_i^{'} \\ .. \\ \mathbf{d}_n \end{bmatrix}$.

Definition 3. An allocation is defined to be a *fixed* allocation when the resources allocated to each agent are fixed over the intervals determined by the different completion times of the agents. The time intervals are given by a sorted order of completion times of the agents (where agents are numbered in increasing order of completion time) $[t_a, \; t_b)$, where $t_a \in \{0, t_1, t_2, ..., t_n\}$ and $t_b = \min_{i \in \mathcal{N}} \{t_i : t_i > t_a\}$.

Proofs omitted from this and subsequent sections can be found in the full version of this paper.

Lemma 1. *Given any resource allocation* $\mathbf{A}(\cdot)$*, there exists an equivalent fixed allocation* $\mathbf{A}^{\text{fixed}}(\cdot)$ *where all the agents have the same costs/utilities.*

Hence, without loss of generality, we will consider only fixed allocations for the remainder of the paper.

Dominant Resource Fairness with Work

The DRF-W mechanism is an adaptation of the DRF mechanism (Ghodsi et al. 2011; Parkes, Procaccia, and Shah 2015) to our setting. The DRF mechanism generalizes max-min fairness to settings with multiple goods and Leontief utilities. It terms the resource satisfying, $r^* = \arg\max_r d_{ir}$ the *dominant resource* of an agent *i*. The share A_{ir^*} of dominant resource allocated to agent *i* is its *dominant share*. The DRF mechanism chooses the allocation that equalizes the dominant shares of all the agents and is efficiently computable even at large scale (Kash, O'Shea, and Volos 2018).

The mechanism $f^{\text{DRF-W}}$ initially allocates resources to all the agents using the same resource allocation policy used for DRF until one of the agents completes its job. After each agent finishes its work, $f^{\text{DRF-W}}$ mechanism reruns DRF on the remaining agents. Thus, at time *t* the DRF-W allocation assigns each agent $\lambda(t)$ of the agent's dominant resource and corresponding amounts of the other resources where $\lambda(t)$ is the solution to the linear program,

$$\max \lambda(t)$$
$$\text{subject to} \sum_{i \in \mathcal{N}(t)} \lambda(t) \cdot d_{ir} \leq 1, \quad \forall r \in \mathcal{R}.$$

Here, $\mathcal{N}(t)$ denotes the set of agents that have not yet completed at time *t*.

We illustrate the DRF-W mechanism with an example.

Example 1. Consider a set of two agents where one agent requires 1 GB RAM (resource r_1) and 0.5 Mbs network bandwidth (resource r_2) and the other agent requires 0.25 GB RAM and 1 Mbs network bandwidth. Both agents have one unit of task to complete. In our notation this becomes

$$\mathbf{D} = \begin{bmatrix} 1 & \frac{1}{2} \\ \frac{1}{4} & 1 \end{bmatrix}, \quad \mathbf{K} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \text{and } \mathbf{W} = \begin{bmatrix} 1 & \frac{1}{2} \\ \frac{1}{4} & 1 \end{bmatrix}.$$

The DRF-W allocation at the first interval is obtained by solving the following linear program,

$$\begin{aligned} \max \quad & \lambda \\ \text{subject to} \quad & \lambda \cdot 1 + \lambda \cdot \frac{1}{4} \leq 1, \\ & \lambda \cdot \frac{1}{2} + \lambda \cdot 1 \leq 1 \end{aligned}$$

where λ is the dominant share of both the agents in the first time period. Solving the above linear program gives $\lambda = \frac{2}{3}$, and the DRF-W allocations as

$$\mathbf{A}_1^{\text{DRF-W}}(t) = \langle \frac{2}{3}, \frac{1}{3} \rangle, \quad \forall t \in [0, \frac{3}{2}]$$
$$\mathbf{A}_2^{\text{DRF-W}}(t) = \langle \frac{1}{6}, \frac{2}{3} \rangle, \quad \forall t \in [0, \frac{3}{2}]$$

Both the agents complete at time $\frac{3}{2}$.

As discussed in the introduction, DRF-W is not Pareto optimal. When compared to the LCP solution given in Example 2 in the next section, this example provides another illustration of this phenomenon. However, DRF-W is weakly Pareto optimal, in that it is not possible for an alternate allocation to make all agents strictly better off. The following theorem shows that it satisfies this as well as several other desiderata.

Theorem 1. *DRF-W satisfies weak-PO, SI, EF, and SP.*

Least Cost Product Mechanism

The Least Cost Product (LCP) mechanism chooses resource allocations such that the product of the costs for the allocations given to agents is the minimum. That is,

$$f^{\text{LCP}}(\mathbf{W}) \in \arg\min_{\mathbf{A}(\cdot)} \prod_{i \in \mathcal{N}} c_i(\mathbf{A}_i(\cdot)).$$

Example 2. We illustrate the LCP mechanism with the same example used in the previous section. Consider the same instances of $\mathbf{D}$, $\mathbf{K}$, and $\mathbf{W}$ used in Example 1 described in the previous section. The completion time of the first agent is given by $t_1 = \frac{k_1}{\lambda_{1,1}}$, where $\lambda_{1,1}$ is the dominant share of the first agent. The completion time of the second agent is expressed in terms of the work remaining for the second agent to complete after the completion of the first agent. In the upcoming subsection, we give a more detailed description of expressing the completion time of an agent in Equation (1). The cost of the second agent is $t_2 = \frac{k_2}{\lambda_{2,2}} - \frac{k_1}{\lambda_{1,1}} \cdot \frac{\lambda_{2,1}}{\lambda_{2,2}} + \frac{k_1}{\lambda_{1,1}}$, where $\lambda_{2,1}$ and $\lambda_{2,2}$ are the dominant shares of the second agent at the first and second time intervals respectively. The LCP allocation is obtained by solving the following optimization problem,

$$\begin{aligned} \min \quad & \frac{1}{\lambda_{1,1}} \cdot \left(1 - \frac{\lambda_{2,1}}{\lambda_{1,1}} + \frac{1}{\lambda_{1,1}}\right) \\ \text{subject to} \quad & \lambda_{1,1} \cdot 1 + \lambda_{2,1} \cdot \frac{1}{4} \leq 1, \\ & \lambda_{1,1} \cdot \frac{1}{2} + \lambda_{2,1} \cdot 1 \leq 1 \end{aligned}$$

where we have substituted $\lambda_{2,2} = 1$ since there are only two agents. Solving the above optimization problem gives us $\lambda_{1,1} = \frac{6}{7}$, $\lambda_{2,1} = \frac{4}{7}$. The LCP allocations are as follows,

$$\mathbf{A}_1^{\text{LCP}}(t) = \langle \frac{6}{7}, \frac{3}{7} \rangle, \quad \forall t \in [0, \frac{7}{6})$$
$$\mathbf{A}_2^{\text{LCP}}(t) = \begin{cases} \langle \frac{1}{7}, \frac{4}{7} \rangle, & \forall t \in [0, \frac{7}{6}) \\ \langle \frac{1}{4}, 1 \rangle, & \forall t \in [\frac{7}{6}, \frac{3}{2}] \end{cases}$$

In comparison to the DRF-W allocation in Example 1, this example illustrates the opportunity for improving the welfare of one agent at no cost to the other.

Before characterizing the behavior of LCP, we observe that it satisfies Pareto optimality and sharing incentives.

Theorem 2. *The LCP mechanism satisfies PO.*

Proof. Suppose for contradiction the allocation output $\mathbf{A}^{\text{LCP}}(\cdot)$ from the LCP mechanism is not PO. Then, there exists an alternative allocation $\mathbf{A}'(\cdot)$ that decreases the cost of at least one agent without increasing the cost of any other agent. This implies $\prod_{i \in \mathcal{N}} c_i(\mathbf{A}'_i(\cdot)) < \prod_{i \in \mathcal{N}} c_i(\mathbf{A}_i^{\text{LCP}}(\cdot))$ which contradicts the optimality of the LCP solution. □

Theorem 3. *The LCP mechanism satisfies SI.*

Structure of LCP Allocations

In this subsection, we characterize the structure of LCP allocations. We show that, apart from the possibility of ties, they consist of allocations where on each time interval the allocation is an extreme point of the Pareto frontier. For most instances, this suffices to reduce finding the exact allocation to examining a finite number of cases. This characterization is used for several of our subsequent results and forms the basis of our simulations.

Let $\mathcal{N} = \{1, 2, ..., n\}$ denote the set of agents where the agents are numbered as per their completion times under the LCP mechanism, that is, agent 1 finishes first, agent 2 second, and so on. The cost product of the agents under the LCP mechanism is expressed as, $CP(\mathbf{A}^{\text{LCP}}(\cdot)) = \prod_{i \in \mathcal{N}} t_i$ where t_i denotes the completion time of agent $i \in \mathcal{N}$. Let $\lambda_{i,j}$ denote the dominant share allocated to agent *i* during the time interval where agent *j* completes its work. The completion time of agent 1 is given by, $t_1 = \frac{k_1}{\lambda_{1,1}}$, and for agents $i > 1$ the completion time is expressed recursively as,

$$t_i = \frac{k_i - \sum_{j=1}^{i-1}((t_j - t_{j-1}) \cdot \lambda_{i,j})}{\lambda_{i,i}} + t_{i-1} \quad (1).$$

For example, the completion time of agent 2 is given by,

$$t_2 = \frac{k_2 - t_1 \cdot \lambda_{2,1}}{\lambda_{2,2}} + t_1 = \frac{k_2 - \frac{k_1}{\lambda_{1,1}} \cdot \lambda_{2,1}}{\lambda_{2,2}} + \frac{k_1}{\lambda_{1,1}}.$$

Now, we analyze the cost product of the LCP mechanism $CP(\mathbf{A}^{\text{LCP}}(\cdot))$ by studying the cost product as a function of the parameters of the *i*-th time interval, $CP_i(\lambda_{i,i},\ \lambda_{i+1,i},\ \lambda_{i+2,i},\ \cdots, \lambda_{n,i})$ while taking the other allocation parameters as constants. That is, $CP_i(\lambda) = CP(\mathbf{A}^{\lambda})$, where $\mathbf{A}^{\lambda}$ is the modified LCP allocation with the λ parameters for the *i*-th time interval.

Lemma 2. *$CP_i(\lambda_{i,i},\ \lambda_{i+1,i},\ \lambda_{i+2,i},\ \cdots, \lambda_{n,i})$ is a Quasiconcave function.*

Proof. First, we show that the product of the completion time of an agent with the dominant share allocated to the *i*-th completing agent is affine. Then, we apply this property to transform the cost product to a univariate function. Finally, we show this univariate function is Quasiconcave.

Without loss of generality, when we analyze $CP_i(\lambda_{i,i},\ \lambda_{i+1,i},\ \lambda_{i+2,i},\ \cdots, \lambda_{n,i})$ we assume that agents $\{1, 2, , \cdots, i-1\}$ have completed their tasks resulting in completion times of agents $t_1, t_2, \cdots, t_{i-1}$ to be regarded as constants. In the i-th time interval, an updated tasks matrix is obtained as $\mathbf{K}^{'}$ where $k_j^{'} = 0$ for agents $1 \leq j \leq i-1$ since the tasks for set of agents $\{1, 2, \cdots, i-1\}$ have already been completed, and $0 \leq k_j^{'} \leq k_j$ for agents $i \leq j \leq n$. That is, agents in $\mathcal{N}^{'} = \{i, i+1, \cdots, n\}$ have not completed their tasks but may have made some progress in earlier intervals.

Let $u_j = t_j \cdot \lambda_{i,i}$. We show by induction that u_j is affine. In the base case we have $j = i$, for which $u_i = t_i\lambda_{i,i} = k_i^{'} + \lambda_{i,i}t_{i-1}$. This is affine as t_{i-1} is a constant. Assume $u_i, u_{i+1}, \cdots, u_{j-1}$ are affine functions of $\lambda_{i,i};\ \lambda_{i,i}, \lambda_{i+1,i};\ \cdots;\ \lambda_{i,i}, ..., \lambda_{j-1,i}$ respectively. We now show that u_j is an affine function of $\lambda_{i,i}, \lambda_{i+1,i}, ..., \lambda_{j-1,i}, \lambda_{j,i}$.

$$\begin{aligned}
u_j &= t_j \cdot \lambda_{i,i} \\
&= \left(\frac{k_j^{'} - \sum_{a=i}^{j-1}(t_a - t_{a-1}) \cdot \lambda_{j,a}}{\lambda_{j,j}} + t_{j-1} \right) \cdot \lambda_{i,i} \\
&= \frac{k_j^{'}}{\lambda_{j,j}}\lambda_{i,i} - \frac{1}{\lambda_{j,j}} \sum_{a=i}^{j-1}(u_a - u_{a-1}) \cdot \lambda_{j,a} + u_{j-1} \\
&= \frac{k_j^{'}}{\lambda_{j,j}}\lambda_{i,i} - \frac{k_i^{'}}{\lambda_{j,j}}\lambda_{j,i} - \frac{\sum_{a=i+1}^{j-1}(u_a - u_{a-1})\lambda_{j,a}}{\lambda_{j,j}} + u_{j-1}
\end{aligned}$$

The first term is linear with respect to $\lambda_{i,i}$. The second term is linear with respect to $\lambda_{j,i}$. The last term of the expression is u_{j-1} which is affine with respect to $\lambda_{i,i}, \lambda_{i+1,i}, \cdots, \lambda_{j-1,i}$. The expression in the third term, results in a linear combination of $u_i, u_{i+1}, \cdots, u_{j-1}$. Hence, u_j is an affine function of $\lambda_{i,i}, \lambda_{i+1,i}, \cdots, \lambda_{j,i}$.

Let us call $\boldsymbol{\lambda} = (\lambda_{i,i}, \lambda_{i+1,i}, ..., \lambda_{n,i})$ a valid allocation of $n - i + 1$ agents in the i-th time interval if $\lambda_{j,i} \geq 0$ for all $1 \leq i \leq j \leq n$, and $\sum_{j \in \mathcal{N}'} \lambda_{j,i} \cdot d_{j,r} \leq 1$ for all $r \in \mathcal{R}$.

Let $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}^{'}$ represent any two valid allocations of the $n - i + 1$ agents in the i-th time interval. We transform the cost product during the i-th time interval into a univariate function $f_{\text{CP}_i}(\theta)$ for $0 \leq \theta \leq 1$. Let,

$f_{\text{CP}_i}(\theta) = CP_i(\theta \cdot \boldsymbol{\lambda} + (1-\theta) \cdot \boldsymbol{\lambda}^{'})$; or

$$f_{\text{CP}_i}(\theta) = \prod_{j \in \mathcal{N}'} \frac{u_j(\theta \cdot \boldsymbol{\lambda} + (1-\theta) \cdot \boldsymbol{\lambda}^{'})}{(\theta \cdot \lambda_{i,i} + (1-\theta) \cdot \lambda_{i,i}^{'})}$$

Now, $CP_i(\boldsymbol{\lambda})$ is Quasiconcave if and only if the univariate function $f_{\text{CP}_i}(\theta)$ is Quasiconcave for all such $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}^{'}$. Since, u_j is an affine function of $\boldsymbol{\lambda}$,

$$\begin{aligned}
f_{\text{CP}_i}(\theta) &= \prod_{j \in \mathcal{N}'} \frac{\theta u_j(\boldsymbol{\lambda}) + u_j(\boldsymbol{\lambda}^{'}) - \theta u_j(\boldsymbol{\lambda}^{'})}{\theta\lambda_{i,i} + \lambda_{i,i}^{'} - \theta\lambda_{i,i}^{'}} \\
&= \prod_{j \in \mathcal{N}'} \frac{\theta a_j + b_j}{\theta a_0 + b_0} \qquad (2)
\end{aligned}$$

where the following substitutions are made at Equation (2), $a_j = u_j(\boldsymbol{\lambda}) - u_j(\boldsymbol{\lambda}^{'})$, $b_j = u_j(\boldsymbol{\lambda}^{'})$, $a_0 = \lambda_{i,i} - \lambda_{i,i}^{'}$, and $b_0 = \lambda_{i,i}^{'}$. By construction, $b_j, b_0 \geq 0$. Without loss of generality, we assume $a_j \geq 0$.

We analyze the logarithm of $f_{\text{CP}_i}(\theta)$ instead of $f_{\text{CP}_i}(\theta)$ since Quasiconcavity is preserved under monotonic transformations. Lemma 3 below shows $\log(f_{\text{CP}_i}(\theta))$ is Quasiconcave, so $CP_i(\boldsymbol{\lambda})$ is a Quasiconcave function. □

Lemma 3. *Let $f(\theta) = \prod_{j \in \mathcal{N}'} \frac{\theta a_j + b_j}{\theta a_0 + b_0}$. Suppose $f(\theta) > 0$ for $\theta \in [0, 1]$; $a_0, b_0 \geq 0$; and $b_j \geq 0$ for $j \in \mathcal{N}^{'} \subseteq \mathcal{N}$. Then, $\log(f(\theta))$ is Quasiconcave on $[0, 1]$.*

We will show that the set of allocations, considered for each period is convex. Since the minimum of a function on a convex set is at one of the extreme points, Lemma 2 tells us we can restrict to examining these, of which there are a finite number (holding the parameters for other periods fixed). We know that LCP allocations are Pareto optimal from Theorem 2. Hence, we must look at the extreme points that lie on the Pareto frontier for every combination from the set of $\mathcal{N}$ agents. The Pareto frontier is determined by the capacity of a resource, $\sum_{i \in \mathcal{N}} \lambda_i \cdot d_{ir} \leq 1, \quad \forall r \in \mathcal{R}$ where λ_i denotes the instantaneous allocation of agent *i*, and d_{ir} corresponds to the demand of agent *i* for resource *r*. The Pareto frontier is the boundary defined by at most *m* hyperplanes where each hyperplane corresponds to a resource $r \in \mathcal{R}$ getting saturated. The extreme points are defined by the relevant intersections of these hyperplanes, i.e. points where at least two resources are saturated as well as points where a single agent is allocated. This characterization is similar in spirit to the result obtained by Bogomolnaia et al. (2017) that an allocation is *competitive* when dividing a set of bads if and only if the utility profile is negative and the allocation is a critical point of the product of the absolute values of the utilities in the negative efficiency frontier.

However, our setting has one additional complication not present in their setting. In addition to resource constraints, since our optimization is based on optimizing on one period while holding the others constant we also have constraints to ensure that all agents finish in the correct order. There may be extreme points of the feasible set where these constraints are tight, which represent ties. We give such an example in the full version of this paper.

Theorem 4. *An LCP allocation $\mathbf{A}^{\text{LCP}}(\cdot)$ consists of at most n time interval allocations. Unless two agents tie in completion time, the allocation of agents in each interval is an extreme point of the Pareto frontier.*

Proof. Let the LCP solution be given, with agents ordered by their completion time. Since this minimizes the cost product, it must also do so when we only optimize over the allocation in one time period, holding all the others fixed.

We know from Lemma 2 that the cost product function at the i-th time interval expressed as a function of the allocation of $n - i + 1$ agents left to finish their work $CP_i(\lambda_{i,i},\ \lambda_{i+1,i},\ \cdots,\ \lambda_{n,i})$ is Quasiconcave. The minimum of a Quasiconcave function over a closed convex set is attained at the extreme points of that closed convex set.

Our optimization problem over $(\lambda_{i,i},\ \lambda_{i+1,i},\ \cdots,\ \lambda_{n,i})$ has three types of constraints. The first two, $\lambda_{j,i} \geq 0$ and $\sum_{j\in\mathcal{N}} \lambda_j \cdot d_{jr} \leq 1$ are non-strict linear inequalities, so their intersection is a closed convex set. The third ensures that agent j is in fact the j-th agent to finish, i.e., $t_j \leq t_{j+1}$ for all j. As shown in the proof of Lemma 2, the u_j (which are a monotone transformation of the t_j) are affine functions, so each of these constraints defines a closed convex set. Our feasible set represents the intersection of these constraints as well and hence, it is closed and convex.

We also know from Theorem 2 that the LCP allocations are Pareto optimal. Thus, we only need to consider extreme points which lie on the Pareto frontier. If these extreme points are defined only by the first two types of constraints, then they are extreme points of the Pareto frontier itself. Otherwise, they include a constraint of the third type being tight, i.e. there are two agents with a tie in completion time. □

Lemma 4. *The LCP mechanism violates SP.*

Envy-freeness of LCP

In all prior settings we are aware of with divisible goods, solutions based on maximizing Nash welfare have been envy-free (Varian 1974; Moulin 2004; Bogomolnaia et al. 2017, 2019; Goel, Hulett, and Plaut 2019). Even in non-divisible settings, it has strong envy-freeness properties that have led it to be described as "unreasonably fair" (Caragiannis et al. 2016). Surprisingly, we show in our setting that the LCP solution is not envy-free, even in expectation.

Lemma 5. *The LCP mechanism does not satisfy EF (even in expectation) when the resource allocation involves three or more agents and two or more resources.*

Consider an example of three agents requiring two resources that have the demand matrix $\mathbf{D}$, and $\mathbf{K}$ as follows,

$$\mathbf{D} = \begin{bmatrix} 1 & 1 \\ 1 & \epsilon \\ \epsilon & 1 \end{bmatrix},\ \mathbf{K} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & k_3 \end{bmatrix},$$

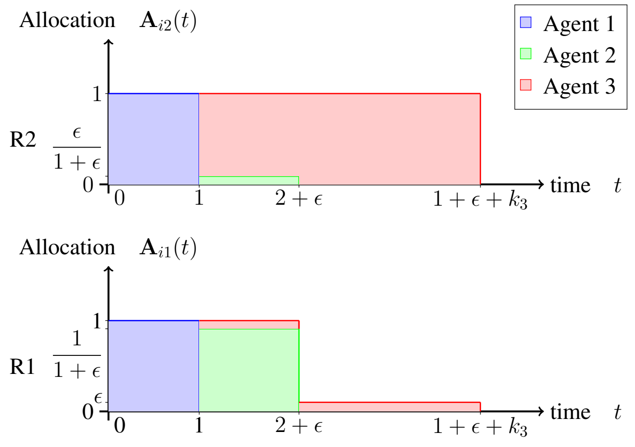

Figure 1: Example where LCP has envy

where $\epsilon << 1$ and $k_3 > 2$.

Figure 1, shows the form of the LCP allocation for the above demands $\mathbf{D}$, and $\mathbf{K}$. In this example, the first and second agents have the same dominant resource and are required to complete the same amount of work. Also, the first agent demands more of the second resource compared to the second agent while, the third agent has a different dominant resource when compared to the second agent. As illustrated by Figure 1, the LCP mechanism would be lowering the cost product by first completing the allocation of the first agent and then allowing the second and third agents to share their allocations. This allocation is efficient but the second agent would envy the first agent. A more detailed description of the example is given in the full version of this paper.

While these examples rule out envy-freeness in general, they seem to be relatively rare in randomly generated examples as our simulations show.

The requirement of three agents and two goods is tight. We show in the full version that with only two agents or a single resource the LCP allocation is EF in expectation.

Lemma 6. *For single resource allocations, the LCP mechanism allocates the resource in the form of the Shortest Job First (SJF) and the allocations are envy-free in expectation. The LCP allocations are envy-free in expectation when allocating multiple resources for two agents.*

Simulation Results

We simulate the LCP and DRF-W mechanisms on randomly generated problem instances in order to better understand the trade-off between fairness and efficiency. The simulation varies the number of agents from 2 to 5. For each number of agents, 2000 examples were generated. The number of resources for each example was chosen uniformly at random between 1 to 10. The demand vector of an agent was generated using a uniform distribution on $(0.0,\ 1.0]$. The demand vector was then normalized for each agent. The amount of work k_i required for agent i to complete was chosen uniformly at random from $(0.0,\ 100.0]$. Using uniform randomness tends to make it harder for the LCP mechanism when compared to more realistic distributions, since if some

Evaluation metrics	Mechanism	Number of agents n			
		$n=2$	$n=3$	$n=4$	$n=5$
Envy-freeness	LCP-X	100 %	99.95 %	99.95 %	99.90 %
	DRF-W	100 %	100%	100 %	100%
Sharing-Incentives	LCP-X	100 %	100%	100 %	100 %
	DRF-W	100 %	100 %	100 %	100 %
Lower Makespan	LCP-X	2.1 %	3.6 %	4.7 %	5.1 %
	DRF-W	58.3 %	73.6 %	76 %	78.6 %
	LCP-X equals DRF-W	39.6 %	23.4 %	19.3 %	16.3 %
Lower mean completion time	LCP-X	95.65 %	99.3 %	100 %	100 %
	DRF-W	4.35 %	0.7 %	0 %	0 %
LCP-X Pareto dominates DRF-W		39.6 %	24 %	20.3 %	18.15 %

Table 1: Comparison of LCP-X and DRF-W on 2000 instances for each n

resources are globally more desired than others or if there are large disparities in k_i then there are effectively fewer resources or agents to consider.

The DRF-W allocation is straightforward to compute as a DRF allocation among agents over each interval. Theorem 4, characterizes the structure of LCP allocations, but the possibility of ties makes an exact computation challenging. Instead, we consider only the main branch of the characterization, where in each period the allocation is an extreme point of the Pareto frontier. We refer to the algorithm that considers only such candidates as LCP-X. With this restriction, we can compute the LCP-X allocation by enumeration, which is polynomial in the number of resources but exponential in the number of agents. While LCP-X does not immediately inherit the properties of LCP, the LCP-X allocation is based on the same principle as the LCP mechanism and is substantially easier to compute because the LCP-X minimizes over a restricted set that excludes certain types of ties. Our simulations show that despite this restriction, the LCP-X generally satisfies the same desiderata as the LCP.

Table 1 summarizes the results of the comparisons between the DRF-W allocation and the LCP-X allocation. Consistent with our theoretical results for LCP, the LCP-X allocations always satisfied SI and were EF with two agents. Rare instances showed envy for LCP-X with more than two agents, but upon inspection these were instances where LCP would have envy as well. This suggests that, at least for randomly generated instances, we should expect the LCP mechanism to almost always satisfy EF. Our use of a continuous distribution for the amount of work each agent has ensured that ties in k_i did not occur. Thus, there were no instances that would have satisfied EF in expectation but not EF.

We also compare DRF-W and LCP-X in terms of their makespan and average completion time, metrics commonly used in the systems settings which inspired our work (Grandl et al. 2016). The makespan is the completion time of the last agent to complete its work and can be interpreted as a fairness property. We observe that the DRF-W allocation typically has a lower makespan than the LCP-X allocation. Since each agent makes equal progress in the DRF-W allocation, with more agents the makespan of the DRF-W mechanism decreases at a faster rate compared to the LCP mechanism. In the instances where the LCP-X allocation had a lower makespan, the LCP-X shares more aggressively, slightly slowing down the initial agents relative to the DRF-W allocation while speeding up the later agents.

The mean completion time of an agent represents an estimate of how fast an agent is expected to complete its work under the allocation and is an efficiency property. For most cases, the LCP-X allocation has a lower mean completion time because of its higher efficiency. In the examples where the DRF-W allocation had a lower mean completion time, we found that the LCP-X allocation takes the form of the shortest job first allocation whereas the sharing of the DRF-W allocation does a better job at minimizing the makespan thereby lowering the mean completion time.

Finally, we examined if the LCP-X allocation Pareto dominated the DRF-W allocation. From Table 1, we see that the percentage of instances where the LCP-X allocation Pareto dominates the DRF-W allocation decreases as the number of agents increase from $n = 2$ to $n = 5$. This decrease is largely accounted for the increase in the percentage of instances the DRF-W allocation has a lower makespan than the LCP-X allocation.

Conclusion

We studied fair division for agents with limited demands. DRF-W allocations exhibit several fairness properties but are in general inefficient. LCP allocations are highly efficient but have relatively weak fairness properties. We believe this is a reasonable trade-off because in system settings (e.g. Grandl et al. (2016), Hindman et al. (2011), Vavilapalli et al. (2013)), the primary objection to DRF is its lack of efficiency. In contrast, LCP yields substantial efficiency gains and in simulations still appears to generally be fair despite the existence of non-EF examples. For future work, based on our observation that there is room to improve the efficiency of DRF at essentially no cost to fairness which was motivated by Grandl et al. (2016), we plan to explore LCP on jobs composed of subtasks with a DAG structure. Another direction would be to investigate a market interpretation of the LCP similar to the works of Goel, Hulett, and Plaut (2019), and the Eisenberg-Gale convex program. One issue here is that usually EF can be shown from the convex program but we know that the LCP does not satisfy EF.

References

Arrow, K.; and Intriligator, M. 2000. Handbook of mathematical economics. Technical report, Elsevier.

Bogomolnaia, A.; Moulin, H.; Sandomirskiy, F.; and Yanovskaia, E. 2019. Dividing bads under additive utilities. *Social Choice and Welfare* 52(3): 395–417.

Bogomolnaia, A.; Moulin, H.; Sandomirskiy, F.; and Yanovskaya, E. 2017. Competitive division of a mixed manna. *Econometrica* 85(6): 1847–1871.

Caragiannis, I.; Kurokawa, D.; Moulin, H.; Procaccia, A. D.; Shah, N.; and Wang, J. 2016. The Unreasonable Fairness of Maximum Nash Welfare. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, EC '16, 305–322. New York, NY, USA: Association for Computing Machinery. ISBN 9781450339360. doi:10.1145/2940716.2940726. URL https://doi.org/10.1145/2940716.2940726.

Conitzer, V.; Freeman, R.; and Shah, N. 2017. Fair Public Decision Making. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, EC '17, 629–646. New York, NY, USA: Association for Computing Machinery. ISBN 9781450345279. doi:10.1145/3033274.3085125. URL https://doi.org/10.1145/3033274.3085125.

Conitzer, V.; Freeman, R.; Shah, N.; and Vaughan, J. W. 2019. Group fairness for the allocation of indivisible goods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1853–1860.

Friedman, E.; and Henderson, S. 2003. Fairness and efficiency in processor sharing protocols to minimize sojourn times. In *Proceedings of ACM SIGMETRICS*, 229–337.

Ghodsi, A.; Zaharia, M.; Hindman, B.; Konwinski, A.; Shenker, S.; and Stoica, I. 2011. Dominant Resource Fairness: Fair Allocation of Multiple Resource Types. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, 323–336. USA: USENIX Association.

Goel, A.; Hulett, R.; and Plaut, B. 2019. Markets Beyond Nash Welfare for Leontief Utilities. *Web and Internet Economics* .

Grandl, R.; Chowdhury, M.; Akella, A.; and Ananthanarayanan, G. 2016. Altruistic Scheduling in Multi-Resource Clusters. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 65–80. Savannah, GA: USENIX Association. ISBN 978-1-931971-33-1. URL https://www.usenix.org/conference/osdi16/technical-sessions/presentation/grandl_altruistic.

Hindman, B.; Konwinski, A.; Zaharia, M.; Ghodsi, A.; Joseph, A. D.; Katz, R.; Shenker, S.; and Stoica, I. 2011. Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, 295–308. USA: USENIX Association.

Kash, I.; Procaccia, A. D.; and Shah, N. 2014. No agent left behind: Dynamic fair division of multiple resources. *Journal of Artificial Intelligence Research* 51: 579–603.

Kash, I. A.; O'Shea, G.; and Volos, S. 2018. DC-DRF: Adaptive multi-resource sharing at public cloud scale. In *Proceedings of the ACM Symposium on Cloud Computing*, 374–385.

Moulin, H. 2004. *Fair Division and Collective Welfare*, volume 1 of *MIT Press Books*. MIT Press, 1 edition. ISBN 0262633116.

Moulin, H. 2019. Fair division in the internet age. *Annual Review of Economics* 11: 407–441.

Parkes, D. C.; Procaccia, A. D.; and Shah, N. 2015. Beyond dominant resource fairness: Extensions, limitations, and indivisibilities. *ACM Transactions on Economics and Computation (TEAC)* 3(1): 3.

Varian, H. R. 1974. Equity, envy, and efficiency. *Journal of Economic Theory* 9(1): 63 – 91. ISSN 0022-0531. doi:https://doi.org/10.1016/0022-0531(74)90075-1. URL http://www.sciencedirect.com/science/article/pii/0022053174900751.

Vavilapalli, V. K.; Murthy, A. C.; Douglas, C.; Agarwal, S.; Konar, M.; Evans, R.; Graves, T.; Lowe, J.; Shah, H.; Seth, S.; et al. 2013. Apache hadoop yarn: Yet another resource negotiator. In *Proceedings of the 4th annual Symposium on Cloud Computing*, 1–16.