Journal of Artificial Intelligence Research 78 (2023) 1069-1109 Submitted 06/2023; published 12/2023

Improving Resource Allocations by Sharing in Pairs

Robert Bredereck ROBERT.BREDERECK@TU-CLAUSTHAL.DE
Institut für Informatik, Algorithm Engineering,
Humboldt-Universität zu Berlin, Berlin, Germany;
Institut für Informatik,
TU Clausthal, Clausthal-Zellerfeld, Germany

Andrzej Kaczmarczyk ANDRZEJ.KACZMARCZYK@AGH.EDU.PL
AGH University of Krakow, Kraków, Poland;
Faculty IV, Algorithmics and Computational Complexity,
Technische Universität Berlin, Berlin, Germany

Junjie Luo JJLUO1@BJTU.EDU.CN
School of Mathematics and Statistics,
Beijing Jiaotong University, Beijing, China

Rolf Niedermeier ROLF.NIEDERMEIER@TU-BERLIN.DE
Florian Sachse SACHSE.FLORIAN@GMAIL.COM
Faculty IV, Algorithmics and Computational Complexity,
Technische Universität Berlin, Berlin, Germany

Abstract

Given an initial resource allocation, where some agents may envy others or where a different distribution of resources might lead to a higher social welfare, our goal is to improve the allocation *without* reassigning resources. We consider a sharing concept allowing resources being shared with social network neighbors of the resource owners. More precisely, our model allows agents to form pairs which then may share a limited number of resources. Sharing a resource can come at some costs or loss in utility. To this end, we introduce a formal model that allows a central authority to compute an optimal sharing between neighbors based on an initial allocation. Advocating this point of view, we focus on the most basic scenario where each agent can participate in a bounded number of sharings. We present algorithms for optimizing utilitarian and egalitarian social welfare of allocations and for reducing the number of envious agents. In particular, we examine the computational complexity with respect to several natural parameters. Furthermore, we study cases with restricted social network structures and, among others, devise polynomial-time algorithms in path- and tree-like (hierarchical) social networks.

1. Introduction

The fair allocation of resources undoubtedly is a key challenge for modern societies and economies. Applications can be found in such diverse fields as cloud computing, food banks, or managing carbon loads in the context of global warming. Naturally, this topic received high attention in the scientific literature. This also holds true for the special case of indivisible resources (Bouveret et al., 2016), which we concentrate on here. Moreover, we take into account the role of social networks built by agents, a growing line of research (Abebe et al., 2017; Bei et al., 2017; Bouveret et al., 2017; Bredereck et al., 2022b; Chevaleyre et al., 2017; Beynier et al., 2019; Lange & Rothe, 2019; Huang & Xiao, 2020). We bring one further new

aspect into this scenario, reflecting the increasing relevance of "sharing economies" (Belk et al., 2019; Schor & Cansoy, 2019), where agents share resources in a peer-to-peer fashion. Resources to share may be almost everything, for instance, knowledge, machines, time, or natural resources.

Our sharing scenario takes into account the constraints of sharing resources between agents expressed by a network. Given an agent network, *two* adjacent agents may share the very same resource, thus increasing the utility of the resource allocation for at least one of them (assuming positive utility for each resource). We assume the sharing process to be organized and decided by a central authority (for example, a team leader, an executive officer, or a public institution). To get started with this new setting, we focus on a very basic scenario. That is, in our model only two neighbors may share *and*, reflecting the (very human) principle of protection of acquired possession, no agent shall lose its already allocated resources. This conservative principle naturally makes sharing easier to implement, keeping "restructuring costs" lower, and it may even help to "keep peace" among agents. Moreover, it sometimes comes very naturally as depicted in the subsequent knowledge sharing example.

Importantly, we also take account of the social relationships between agents. Consider an executive officer and an employee of a company. It is conceivable that the executive officer compares herself (in any way) to the employee, but the opposite is not true as the employee does not even know the executive officer. In our scenario, we use directed social networks, to capture this kind of asymmetric social relations and model various social structures of the agents.

We apply our sharing concept to improve egalitarian or utilitarian welfare. Besides, we also focus on a fundamental fairness criterion, envy-freeness. Albeit appealing and highly desirable, guaranteeing envy-freeness for every agent is not always possible to achieve (consider one indivisible resource and three agents desiring it). We exert our model to tackle this problem. Respecting the social relationships of the agents, we aim at improving a given allocation by decreasing the number of envious agents through feasible resource sharing.

Knowledge Sharing Before becoming more specific about our model, let us first introduce an example related to knowledge sharing. Assume that agents are employees of a company, each having a bundle of qualifications. An agent may "envy" another agent because the other agent has overall qualification of higher value. The central authority wants to improve the situation by building teams of two agents where, due to a daily extensive cooperation, one teaches the other the missing qualification. For instance, a realization of this is the concept of *pair programming* that also has other benefits besides knowledge sharing (Williams, Kessler, Cunningham, & Jeffries, 2000).

1.1 Model of Sharing Allocation

Roughly speaking, our model is as follows (see Section 3 for formal definitions). The input is a set of agents and a set of indivisible resources initially assigned to the agents. Typically, every agent may be assigned several resources. Each agent has an individual utility value for each resource. The general goals are to decrease the overall degree of envy, to increase the sum of "utility scores" of all agents, or to increase the minimal "utility score" among

all agents. Importantly, the only way an agent can improve its individual "utility score" is by participating in a sharing with another agent.

The model variant on which we mostly focus assumes that the utility value of a resource does not decrease when it is shared. This approach is justified when the burden of sharing is neutralized by its advantages. Indeed, in our knowledge sharing example a hassle of cooperation is often compensated by a better working experience or higher quality outcomes (as shown by Williams et al., 2000). Note that such complicated mutual dependencies that would be extremely hard to describe formally form a natural field for our approach. Further application examples include irregularly used resources (like printers or computing servers in universities). Here, the coordination with another person is uncritical and splitting the maintenance costs neutralizes the inconvenience of cooperation.

We enrich our model by using two graphs, an undirected sharing graph and a directed attention graph, to model social relations between agents and to govern the following two constraints of our model. The sharing graph models the possibility for two agents to share resources. We assume that only neighbors in the sharing graph can share a resource (a missing qualification in our knowledge sharing example). In practice, the sharing graph might for example reflect whether agents are close enough to each other to be able to share a resource or whether there is no conflict between the time they use resources. With respect to lowering the degree of envy, we assume that agents may only envy their outneighbors in the directed attention graph. This graph-based envy concept has recently been studied by several works in the fair allocation domain (Bredereck et al., 2022b; Aziz et al., 2018; Beynier et al., 2019).

Agents may naturally be conservative in the sense of keeping control and not sharing too much. Furthermore, as in our knowledge sharing example, it might simply be too ineffective to share a qualification among more than two employees simultaneously (due to, e.g., increased communication overhead or additional resources needed). We address this in the most basic way and assume that each resource can be shared to *at most one neighbor* of its owner and an agent can participate in a bounded number of sharings. This strong restriction already leads to tricky algorithmic challenges and fundamental insights. In particular, the model also naturally extends on well-known matching scenarios in a non-trivial way.

There are numerous options to further extend and generalize our basic model, as discussed in Section 6 and in the concluding Section 7. However, keeping our primary model simple, we aim at spotting its fundamental properties influencing the complexity of related computational problems.

1.2 Related Work

To the best of our knowledge, so far the model we consider has not been studied. Since obtaining envy-free allocations is not always possible, there has been work on relaxing the concept of envy. In particular, in the literature *bounded-maximum envy* (Lipton et al., 2004), *envy-freeness up to one good* (Budish, 2011), *envy-freeness up to the least-valued good* (Caragiannis et al., 2019), *epistemic envy-freeness* (Aziz et al., 2018), *maximin share guarantee* (Budish, 2011), and *information withholding* (Hosseini et al., 2020) have been studied. However, these concepts combat nonexistence of allocations that are envy-free by

considering approximate versions of it. By way of contrast, our approach tries to find a way to lessen envy not by relaxing the concept of envy, but rather by enabling a small deviation in the model of indivisible, non-shareable resources. To this end, we make resources shareable (in our basic model by two agents). This approach is in line with a series of recent works which try to reduce envy (i) by introducing small amounts of money (Brustle et al., 2020; Halpern & Shah, 2019; Caragiannis & Ioannidis, 2022), (ii) by donating a small set of resources to charity (Caragiannis et al., 2019; Chaudhury et al., 2021), or (iii) by allowing dividing a small number of indivisible resources (Sandomirskiy & Segal-Halevi, 2022; Segal-Halevi, 2019). In particular, the papers mentioned in point (iii) consider a model of indivisible resources that could be shared by an arbitrary group of agents and where, unlike in our study, each agent only gets a portion of the utility of the shared resources. Contrary to our setting, this model assumes no initial allocation. As a result, an envy-free allocation always exists and the goal is to seek one with a minimum number of shared resources. In contrast, our goal is to improve an *initial* allocation through sharing resources between pairs of agents. Another line of research considers the improvement of allocations by exchanging resources (Chevaleyre, Endriss, & Maudet, 2007; Gourvès, Lesca, & Wilczynski, 2017; Huang & Xiao, 2020). There has been quite some work on bringing together resource allocation and social networks (Abebe et al., 2017; Bei et al., 2017; Bouveret et al., 2017; Bredereck et al., 2022b; Chevaleyre et al., 2017; Beynier et al., 2019; Huang & Xiao, 2020). In particular, the concept of only local envy relations to neighbors in a graph gained quite some attention (Aziz et al., 2018; Beynier et al., 2019; Bredereck et al., 2022b; Eiben, Ganian, Hamm, & Ordyniak, 2023). Modifying existing allocations to maintain fairness has also been studied for divisible resources (Segal-Halevi, 2022) and in online settings with changing agents (Friedman et al., 2015, 2017; Vardi et al., 2022) or resources arriving over time (He et al., 2019).

1.3 Our Contributions

Introducing a new model for (indivisible) resource allocation with agents linked by social networks, we provide a view on improving existing allocations for several measures without, conceivably impossible, reallocations.

We analyze the (parameterized) computational complexity of applying our model to decrease the number of envious agents (Definition 7) as well as to improve the classical social welfare measures (Definition 6): the utilitarian one, following the philosophical foundation by Bentham (1823) and the egalitarian one implementing the social equality desideratum by Rawls (1971). We show that a central authority can (mostly) find a sharing that improves social welfare (measured in both the egalitarian and utilitarian ways) in polynomial time, while decreasing the number of envious agents is NP-hard even if the sharing graph is a clique and the attention graph is a bidirectional clique.

To overcome NP-hardness, we also study the influence of different natural parameters (such as agent utility function values, structural parameters concerning the agent social networks, the number of agents, and the number of resources); Table 1 surveys our results in more detail. We show that the problem is polynomial-time solvable if the underlying undirected graph of the attention graph is the same as the sharing graph and has a constant treewidth (intuitively, the sharing graph is "close to" a tree). We also identify an interesting

Envy-Reducing Sharing Allocation (ERSA)

$\mathcal{G}_s \equiv \mathcal{G}_t$		unanimous utility functions		few agents		few resources		
clique	tree- or pathwidth	$\mathcal{G}_t$ = clique	$\mathcal{G}_s$ = clique	n	$k = 0, \Delta k = 1$	m	#shared resources	bundle size
NP-h Thm. 3	XP, W[1]-h Thm. 6	P Thm. 7	NP-h★ Thm. 8	FPT Thm. 4	p-NP-h Thm. 5	XP, W[1]-h★ Obs. 1, Thm. 8		p-NP-h Thm. 8

Table 1: Results overview for ERSA, where $\mathcal{G}_s \equiv \mathcal{G}_t$ means that the sharing graph ($\mathcal{G}_s$) is the same as the underlying graph of the attention graph ($\mathcal{G}_t$), n is the number of agents, k is the number of envious agents after sharing, Δk is a drop in the number of envious agents, and m is the number of resources. Remarkably, in all hardness results the number of different utility values used by the utility functions is between 2 and 4. The result marked with ★ holds also for the case of indistinguishable resources.

contrast between the roles of the two graphs: When agents have the same utility function, the problem is solvable in polynomial time if the attention graph is a bidirectional clique, while the problem is NP-hard even if the sharing graph is a clique. Finally, we show that the problem is fixed-parameter tractable (FPT) for the parameter number of agents (giving hope for efficient solutions in case of a small number of agents) and polynomial-time solvable (in XP) for a constant number of resources. However, the problem is NP-hard even if the goal is to reduce the number of envious agents from one to zero. We remark that all our hardness results are shown for cases in which the number of different values of utility functions is a small constant (at most 4).

Altogether, our main technical contributions are with respect to exploring the potential to "overcome" the NP-hardness of decreasing the number of envious agents by exploiting several problem-specific parameters.

2. Preliminaries

We mostly only cover the background necessary to understand our proofs and results. However, when appropriate, we refer the reader to recent textbooks for detailed information on the subject.

2.1 Parameterized Complexity

A computational problem is fixed-parameter tractable for some parameter x then it can be solved in time $f(x) \cdot |I|^{O(1)}$, where f is an arbitrary computable function and $|I|$ is the size of the instance—we refer to algorithms guaranteeing such a running time as FPT *algorithms*. If a problem is W[1]-hard, then it is (presumably) not fixed-parameter tractable. Yet, if there is an algorithm running in polynomial-time assuming a constant value of parameter x, then we say that the problem is *in* XP. See recent textbooks for details about the parameterized algorithms approach (Downey & Fellows, 2013; Cygan et al., 2015).

2.2 Graphs

A *directed graph* G consists of a set V of *vertices* and a set $E \subseteq V \times V$ of *arcs* connecting the vertices; we only consider directed graphs *without self-loops*, that is, there are no arcs of form (v, v) for any vertex $v \in V$. A (simple) *undirected graph* $G = (V, E)$ consists of a set V of vertices and a set E of (distinct) size-2 subsets of vertices called *edges*. The *underlying undirected graph* of a directed graph G is the graph obtained by replacing all (directed) arcs with (undirected) edges. By our definition of (simple) undirected graphs, even if a directed graph G contains two opposite arcs (u, v) and (v, u), then its underlying undirected graph has a single edge $\{u, v\}$.

We say an undirected graph $G = (V, E)$ is a *clique* if $E = \binom{V}{2}$ and a directed graph $G = (V, E)$ is a *bidirectional clique* if $E = (V \times V) \setminus \{(v, v) \mid v \in V\}$. For some vertex $v \in V$ of graph $G = (V, E)$, the set $I(v) \subseteq E$ of *incident* arcs (edges) is the set of all arcs (edges) with an endpoint in v.

Associating the edges (or arcs) of a graph $G = (V, E)$ with some weight function $w\colon E \to \mathbb{Q}$, we obtain a *weighted* graph.

2.3 Tree Decompositions and Treewidth

Intuitively, a tree decomposition is a tree-representation of a given graph. Tree decompositions enable designing dynamic programming algorithms that efficiently solve hard problems. Informally speaking, these algorithms extend the techniques for trees to graphs that are similar to trees, encapsulating the exponential blow-up of the running times in a function that depends solely on "how similar" the original graph is to a tree. This similarity is measured by treewidth, an integral measure of a graph, which is 1 when the original graph itself is a tree. We employ this approach (in later sections) and thus obtain FPT algorithms with respect to treewidth. Hence, we put on formal definitions illustrated on Figure 1, which also informally explains the way the dynamic programming approach takes advantage of tree decompositions.

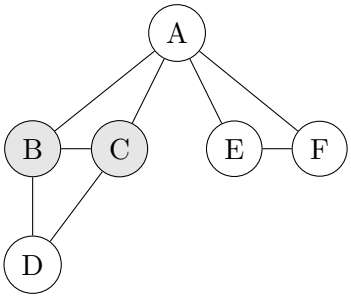
Definition 1. *Given a graph $G = (V, E)$, a* tree decomposition $\mathcal{T}$ *consists of a rooted tree $T = (V', E')$ and a labeling that assigns each* node $v'_i \in V'$ *its* bag $X_i \subseteq V$ *such that:*

1. *For each edge $\{u, v\} \in E$, there is a bag X_j such that $\{u, v\} \subseteq X_j$;*
2. *For each vertex $v \in V$ and the corresponding maximal set $V^* = \{v'_{\ell_1}, v'_{\ell_2}, \ldots, v'_{\ell_t}\} \subseteq V'$ of nodes of T such that, for each $i \in \{\ell_1, \ldots, \ell_t\}$, $v \in X_i$, it holds that nodes in V^* form a connected subtree in T.*

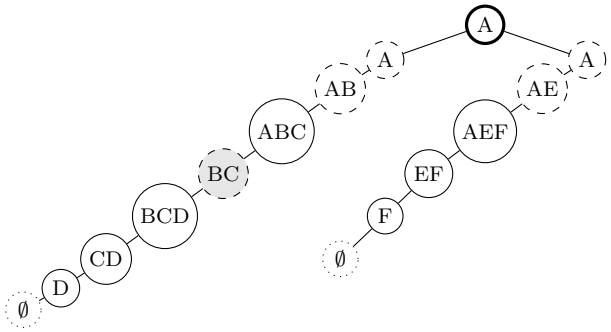
The width *of $\mathcal{T}$ is the size of the largest* bag *decreased by one, and the* treewidth *of G is the minimum width over all possible tree decompositions of G.*

A very specific type of tree decompositions—a nice tree decomposition—is particularly convenient to design dynamic-programming-based algorithms.

Definition 2. *A* tree decomposition $\mathcal{T}$ *with a rooted tree $T = (V', E')$ and the corresponding bags X_1, X_2, …, $X_{|V'|}$ is a* nice tree decomposition *if each node $t_i \in V'$ is of one of the following types:*



(a) An example graph. Note that vertices {B, C} (marked grey) separate vertex D from vertices {A, E, F}.



(b) A nice tree decomposition of the graph. Nodes are labeled with the corresponding bags. The leaf nodes, introduce nodes, forget nodes, and join node are, respectively, dotted, solid, dashed, and in bold. The top node is the root. The grey node corresponds to the separator of vertex D and the rest of the graph.

Figure 1: A graph and its nice tree decomposition. Note that exercising dynamic programming on such a decomposition (applying the bottom-up processing) allows us finding optimal solutions for certain subgraphs of the original graph. For example, as we reach bag {B, C} (marked grey) of the decomposition, vertex D does not appear later at any step of the computation. Indeed, as B and C are separating D from the rest of the graph, the latter cannot influence the situation for the remaining part of the graph.

1. ***Leaf node***, *where* t_i *is a leaf in* T *and* $|X_i| = 1$*;*
2. ***Introduce node***, *where* t_i *has a single child* $t_{i'} \in V'$ *in* T *and* $X_i = X_{i'} \cup \{v\}$ *for some vertex* $v \in V \setminus X_{i'}$*;*
3. ***Forget node***, *where* t_i *has a single child* $t_{i'} \in V'$ *in* T *and* $X_i = X_{i'} \setminus \{v\}$ *for some vertex* $v \in V \cap X_{i'}$*; or*
4. ***Join node***, *where* t_i *has exactly two (distinct) children* $t_{i'} \in V'$ *and* $t_{i''} \in V'$ *in* T *and it holds that* $X_i = X_{i'} = X_{i''}$*.*

Importantly, for every graph G a nice decomposition can be computed in time $f(\mathrm{tw}) \cdot n$, where tw is the treewidth of G (Cygan et al., 2015) and n is the number of vertices in G. We refer to the book by Kloks (1994) for more details on tree decompositions and treewidth.

2.4 Matchings and Matching Problems

Given an undirected graph $G = (V, E)$, a subset M of the edges is a *matching in* G if no pair of edges in M share a vertex. A matching whose cardinality (size) is not smaller than the size of any other matching in G is a *maximum matching*. Analogously, considering a weight of a matching in an undirected weighted graph as the sum of the weights of the matching's

edges, we define *maximum weight* and *minimum weight* matchings. Computational problems related to finding various kinds of matchings are ubiquitous. Polynomial algorithms solving them have been extensively studied for several decades, and are constantly being improved and specialized. Hence, wherever we analyze the running time of our algorithms that depend on these fundamental problems, we express it using the running time of a matching algorithm as a variable. In particular, we will use ρ for the running time of finding a maximum matching and ρ_{w} for the running time of finding a maximum weighted matching. For quite a recent listing of the quickest algorithms solving matching-related problems under different constraints, we refer to the preliminaries of the works of Duan and Pettie (2014) and Duan et al. (2018).

3. Modeling

For a set $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ of *agents* and a set $\mathcal{R} = \{r_1, r_2, \ldots, r_m\}$ of *indivisible resources*, a *(simple) allocation* $\pi\colon \mathcal{A} \to 2^{\mathcal{R}}$ is a function assigning to each agent a collection of resources—a *bundle*—such that the assigned bundles are pairwise disjoint. An allocation is *complete* if every resource belongs to some bundle.

3.1 Sharing

We fix an initial allocation π of resources in $\mathcal{R}$ to agents in $\mathcal{A}$. A *sharing graph* is an undirected graph $\mathcal{G}_{\mathrm{s}}{=}(\mathcal{A},\mathcal{E}_{\mathrm{s}})$ with vertices being the agents; it models possible sharings between the agents. The following definition of sharing says that two agents can only share resources held by one of them.

Definition 3. *Function $\delta_\pi\colon \mathcal{E}_s \to 2^{\mathcal{R}}$ is a* sharing *for some allocation π if for every two agents a_i and a_j, with $\{a_i, a_j\} \in \mathcal{E}_s$, it holds that $\delta_\pi(\{a_i, a_j\}) \subseteq \pi(a_i) \cup \pi(a_j)$.*

An initial allocation π and a corresponding sharing δ_π form a *sharing allocation*.

Definition 4. *A* sharing allocation induced by *allocation π and sharing δ_π is a function $\Pi_\pi^{\delta_\pi}\colon \mathcal{A} \to 2^{\mathcal{R}}$ where $\Pi_\pi^{\delta_\pi}(a) \coloneqq \pi(a) \cup \bigcup_{e \in I(a)} \delta_\pi(e)$.*

Since the initial allocation π is fixed, for brevity, we use δ and Π^δ, omitting π whenever it is not ambiguous. For simplicity, for every agent $a \in \mathcal{A}$, we also refer to $\Pi^\delta(a)$ as a *bundle* of a whenever it does not lead to confusion.

Naturally, each allocation is also a sharing allocation with a trivial "empty sharing." Observe a subtle difference in the intuitive meaning of a bundle of an agent between sharing allocations and (simple) allocations. For sharing allocations, a bundle of an agent represents the resources the agent has access to and can utilize, not only those that the agent possesses (as for simple allocations).

3.2 2-sharing

Definition 3 is very general and only requires that two agents share resources that one of them already has. In particular, Definition 3 allows one agent to share the same resource with many other agents; and does not constrain the number of sharings an agent could participate in. In this paper, we assume that each resource can only be shared by *two*

agents and each agent can participate in at most a bounded number of sharings. We formally express this requirement in Definition 5.

Definition 5. *A* 2-sharing δ *is a sharing where, for each triple of distinct agents* a_i, a_j, *and* a_k, *it holds true that*

$$\Pi^\delta(a_i) \cap \Pi^\delta(a_j) \cap \Pi^\delta(a_k) = \emptyset.$$

A b-bounded 2-*sharing* δ *is a* 2-*sharing in which, for each agent* a, *it holds true that* $\left|\bigcup_{e \in I(a)} \delta(e)\right| \leq b$. *A* simple 2-sharing δ *is a* 1-*bounded* 2-*sharing, i.e., for each agent* a, *it holds true that* $\left|\bigcup_{e \in I(a)} \delta(e)\right| \leq 1$.

Herein, we count the number of sharings an agent participate in by the number of resources shared with other agents (either shared to other agents or received from other agents). Notably, in a simple 2-sharing, each agent can *either share* or *receive* a *single resource*. Thus, every simple 2-sharing can be interpreted as a matching in which each edge is labeled with a shared resource.

3.3 Utility, Welfare Measures, and Fairness Concepts

We assume agents have cardinal, monotonic, and additive utility functions. For an agent a with *utility function* $u\colon \mathcal{R} \to \mathbb{N}_0$ and a bundle $R \subseteq \mathcal{R}$, let $u(R) \coloneqq \sum_{r \in R} u(r)$ be the *value* of R as perceived by a. For agents $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ with corresponding utility functions $u_1, u_2, \ldots, u_n$, we say that their utility functions are *unanimous* if the utility functions of all agents are exactly the same, that is, for each pair $a_i, a_j \in \mathcal{A}$ and for each $r \in \mathcal{R}$ it holds that $u_i(r) = u_j(r)$. The resources are *indistinguishable* if the utility functions are unanimous and the unanimous function gives the same utility for every resource.[1]

Let us fix a sharing allocation Π^δ of resources $\mathcal{R}$ to agents $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ with corresponding utility functions $u_1, u_2, \ldots, u_n$. The *utilitarian social welfare* of Π^δ is

$$\mathrm{usw}(\Pi^\delta) \coloneqq \sum_{i \in [n]} u_i(\Pi^\delta(a_i)).$$

The *egalitarian social welfare* of Π^δ is

$$\mathrm{esw}(\pi) \coloneqq \min_{i \in [n]} u_i(\Pi^\delta(a_i)).$$

Notice that, for now, we assume each agent a_i gets the full utility for all resources in $\Pi^\delta(a_i)$. We will discuss a generalization of this assumption in Section 6.

A directed graph $\mathcal{G}_\mathrm{t} = (\mathcal{A}, \mathcal{E}_\mathrm{t})$ with vertices being the agents is an *attention graph*; it models social relations between the agents. We say that an agent a_i *looks at* another agent a_j if $(a_i, a_j) \in \mathcal{E}_\mathrm{t}$. An agent is *envious* on $\mathcal{G}_\mathrm{t}$ under Π^δ if it prefers, compared to its own bundle, a bundle of another agent it looks at; formally, a_i *envies* a_j if $u_i(\Pi^\delta(a_i)) < u_i(\Pi^\delta(a_j))$

1. The property of being indistinguishable might be also considered without assuming unanimity. In such a case, however, one can speak only about the resources being indistinguishable from a perspective of a single agent. It then can happen that two (or more) agents perceive the resources as indistiguishable but every agent gives the resources a different utility value. We do not take this approach as we do not make use of it in our paper.

and $(a_i, a_j) \in \mathcal{E}_\mathrm{t}$. We denote the set of envious agents in Π^δ as $\mathrm{Env}(\Pi^\delta)$. For a given (directed) attention graph $\mathcal{G}_\mathrm{t}$ over the agents, a sharing allocation is *$\mathcal{G}_t$-envy-free* if no agent envies its out-neighbors.

4. Improving Social Welfare by Sharing

In this section, we study the problem of improving utilitarian (and egalitarian) welfare through sharing, defined as follows. Recall that considering welfare does not require the attention graph, so this graph does not appear in the input of the following problems.

Definition 6. *Given an initial complete allocation π of m resources $\mathcal{R}$ to n agents $\mathcal{A}$, a sharing graph $\mathcal{G}_s$, and a non-negative integer k, b-Bounded Utilitarian Welfare Sharing Allocation (b-UWSA) asks if there is a b-bounded 2-sharing δ such that* $\mathrm{usw}(\Pi^\delta) \geq k$*; b-Bounded Egalitarian Welfare Sharing Allocation (b-EWSA) asks if there is a b-bounded 2-sharing δ such that* $\mathrm{esw}(\Pi^\delta) \geq k$.

We first consider b-UWSA. When $b = 1$, since every simple 2-sharing corresponds to a matching, we can easily reduce 1-UWSA to Maximum Weighted Matching. Thus, 1-UWSA is solvable in polynomial time. When $b > 1$, however, the problem is not just finding a couple of matchings such that the total weight is maximized. Notice that in 2-sharing each resource can only be shared once. Nevertheless, we show that we can still reduce b-UWSA to Maximum Weighted Matching via a more involved reduction.

Theorem 1. *For n agents and m resources, b-*UWSA *is solvable in $O((m + bn)^2) + \rho_w$ time for any $b \geq 1$, where ρ_w is the time needed to find a maximum weight matching in a (weighted undirected) graph with at most $2(m + bn)$ vertices, $4(m + bn)^2$ edges, and the maximum weight equal to the biggest utility that an agent gives to a resource.*

Proof. Given an instance of b-UWSA, we transform it into an instance $I = (G = (V, E), w)$ of Maximum Weighted Matching. For simplicity, we assume that each agent $a_i \in \mathcal{A}$ has at least b resources in the initial allocation π; otherwise we ensure this by adding enough resources that are valued as 0 by all agents. We build instance I with $G = (V, E)$, starting from empty sets V and E, as follows. For each agent $a_i \in \mathcal{A}$ and each resource $r_j \in \mathcal{R}$ such that $r_j \in \pi(a_i)$, we add a vertex v_i^j to V. In addition, for each agent $a_i \in \mathcal{A}$, we add to V exactly $n_i := |\pi(a_i)| - b$ *dummy vertices* $\{\overline{v}_i^1, \overline{v}_i^2, \ldots, \overline{v}_i^{n_i}\}$. Consider a pair of distinct agents $a_x, a_y \in \mathcal{A}$ with $\{a_x, a_y\} \in \mathcal{E}_\mathrm{s}$ and a pair of (distinct) resources $r_p, r_q \in \mathcal{R}$ such that $r_p \in \pi(a_x)$ and $r_q \in \pi(a_y)$. We add to E an edge of weight $\max\{u_x(r_q), u_y(r_p)\}$ between vertices v_x^p and v_y^q from V[2]. We repeat this step for all possible, distinct collections of pairs like the abovementioned one. Finally, for each agent $a_i \in \mathcal{A}$, we connect every vertex $v_i^j \in V$, such that $r_j \in \mathcal{R} \cap \pi(a_i)$, with every dummy vertex $\overline{v}_i^k$, $k \in [n_i]$, related to a_i by a *dummy edge* with weight $W := \max_{a_x \in \mathcal{A}, r_p \in \mathcal{R}} u_x(r_p)$. This finishes the construction of I. Note that the constructed graph G has at most $2(m + bn)$ vertices and $4(m + bn)^2$ edges.

2. Associating the added edge directly with the interpretation that we either share r_q or r_p along the edge between agents a_x and a_y in the sharing graph is tempting but incorrect. The meaning of the added edge is slightly more involved, as we show in the proof of the forward direction.

Notice that for $G = (V, E)$ there always exists a maximum weighted matching that, for each agent $a_i \in \mathcal{A}$, contains n_i dummy edges. Let $P \coloneqq W \sum_{a_i \in \mathcal{A}} n_i$ be the weight of those edges. In the following, we show that there is a b-bounded 2-sharing δ such that $\mathrm{usw}(\Pi^\delta) \geq k$ if and only if graph G admits a matching M with weight $\sum_{e \in M} w(e) \geq k - \mathrm{usw}(\pi) + P$, where $\mathrm{usw}(\pi) \coloneqq \sum_{a_i \in \mathcal{A}} u_i(\pi(a_i))$ is the utilitarian welfare of the initial allocation π.

$\Rightarrow$: Assume there is a b-bounded 2-sharing δ such that $\mathrm{usw}(\Pi^\delta) \geq k$. Based on δ, we can build a matching M with the claimed weight. Intuitively, we do so by including the edges corresponding to each $e \in \mathcal{E}_\mathrm{s}$ with $\delta(e) \neq \emptyset$ together with edges between the remaining non-dummy vertices and all dummy vertices.

Formally, first, we consider each edge $\{a_x, a_y\} \in \mathcal{E}_\mathrm{s}$ such that $\delta(\{a_x, a_y\}) \neq \emptyset$ and each resource $r_p \in \delta(\{a_x, a_y\})$. Without loss of generality, suppose $r_p \in \pi(a_x)$. Let $r_q \in \pi(a_y)$ be an arbitrary resource of a_y that is not shared under δ. Since a_y participates in at most b sharings and a_y has at least b resources in the initial allocation π, for each resource that a_y gets as a result of δ, there is a resource in $\pi(a_y)$ that is not shared under δ. We add edge $\{v_x^p, v_y^q\}$ to M. Note that if there are two resources $r_p, r_{p'} \in \delta(\{a_x, a_y\})$ such that $r_p \in \pi(a_x)$ and $r_{p'} \in \pi(a_y)$, then we, by definition of M, do not add the edge $\{v_x^p, v_y^{p'}\}$ to M (because both $r_p$and $r_{p'}$ are shared under δ). Notice that $w(\{v_x^p, v_y^q\}) \coloneqq \max\{u_x(r_q), u_y(r_p)\} \geq u_y(r_p)$. So, the weight of all edges added to M in this step is at least $\mathrm{usw}(\Pi^\delta) - \mathrm{usw}(\pi)$, which represents the utility gained as a result of sharing δ. Let us consider the vertices related to each agent $a_i \in \mathcal{A}$ after performing the first step. It holds that there are at least n_i unmatched non-dummy vertices (as δ is b-bounded) and exactly n_i unmatched dummy vertices (no dummy vertex was matched in the first step). Hence, we can add n_i dummy edges (all of weight W) to M because each such edge contains one normal vertex and one dummy vertex.

Since $\mathrm{usw}(\Pi^\delta) \geq k$, we have that $\sum_{e \in M} w(e) \geq \mathrm{usw}(\Pi^\delta) - \mathrm{usw}(\pi) + P \geq k - \mathrm{usw}(\pi) + P$, which finishes the argument for this direction.

$\Leftarrow$: Assume there is a matching M in graph G with the claimed weight. Without loss of generality, we assume that M contains n_i dummy edges for every agent a_i. This holds because the weight of the dummy edges is at least that of the non-dummy edges. Then, for each agent $a_i \in \mathcal{A}$, M contains at most b non-dummy edges incident to the vertices related to a_i, that is, $\{v_i^j \mid r_j \in \pi(a_i)\}$. Using these non-dummy edges in M, we construct the corresponding b-bounded 2-sharing δ such that $\mathrm{usw}(\Pi^\delta) \geq k$ as follows. For each non-dummy edge $\{v_x^p, v_y^q\} \in M$, we add r_p into $\delta(a_x, a_y)$ if $u_y(r_p) \geq u_x(r_q)$ and add r_q into $\delta(a_x, a_y)$ otherwise. Since $w(\{v_x^p, v_y^q\}) \coloneqq \max\{u_x(r_q), u_y(r_p)\}$, this increases the utilitarian welfare by exactly $w(\{v_x^p, v_y^q\})$ for each non-dummy edge $\{v_x^p, v_y^q\} \in M$. So we have that

$$\mathrm{usw}(\Pi^\delta) = \mathrm{usw}(\pi) + \sum_{e \in M} w(e) - P \geq k.$$

Moreover, since M contains at most b non-dummy edges incident to the vertices related to each $a_i \in \mathcal{A}$, every a_i participates in at most b sharings in δ. The above finishes the proof of the second direction. □

Next we consider b-EWSA. When $b = 1$, we show in Proposition 1 that we can reduce the problem to Maximum Matching. The idea is to partition all agents into two subgroups according to the target k and build a bipartite graph characterizing whether one agent from

one group can improve the utility of one agent from the other group to k by sharing. Then the problem is just to find a maximum matching in the bipartite graph.

Proposition 1. *For n agents and m resources,* 1-EWSA *is solvable in $O(n^2) + \rho$ time, where ρ is the time needed to find a maximum matching in a bipartite graph with at most n vertices and n^2 edges.*

Proof. Depending on the target k, we partition the set $\mathcal{A}$ of agents into two sets $\mathcal{A}_k^+$ and $\mathcal{A}_k^-$ containing, respectively, the agents with their bundle value under π at least k and smaller than k. Now, we construct a bipartite, undirected graph $G_k = (\mathcal{A}_k^+, \mathcal{A}_k^-, E_k)$. Consider two agents $a_i \in \mathcal{A}_k^+$ and $a_j \in \mathcal{A}_k^-$ that are neighbors in the sharing graph $\mathcal{G}_\mathrm{s}$. An edge $e = \{a_i, a_j\}$ belongs to E_k if a_i can share a resource with a_j to raise the utility of the latter to at least k; formally, there exists a resource $r \in \pi(a_i)$ such that $u_j(\pi(a_j)) + u_j(r) \geq k$.

We claim that there is a simple 2-sharing δ with $\mathrm{esw}(\Pi^\delta) \geq k$ if and only if there is matching M in graph G_k with $|M| \geq |\mathcal{A}_k^-|$. The backward direction is clear according to the construction of G_k. For the forward direction, assume that there is a simple 2-sharing δ with $\mathrm{esw}(\Pi^\delta) \geq k$. Then, we can build a matching M in graph G_k by adding an edge $e = \{a_i, a_j\}$ to M if $\delta(\{a_i, a_j\}) \neq \emptyset$ and either a_i or a_j belongs to $\mathcal{A}_k^-$. Notice that since $\mathrm{esw}(\Pi^\delta) \geq k$, the edge $e = \{a_i, a_j\}$ must be in E_k. As δ is a simple 2-sharing, the constructed M is indeed a matching. Since $\mathrm{esw}(\Pi^\delta) \geq k$, according to the construction of G_k and M, we have that for each $a_j \in \mathcal{A}_k^-$, there exists an agent $a_i \in \mathcal{A}_k^+$ such that $\{a_i, a_j\} \in M$, and hence $|M| \geq |\mathcal{A}_k^-|$. Thus, we just need to check whether the maximum matching in graph G_k has size at least $|\mathcal{A}_k^-|$. □

In constrast to b-UWSA, it turns out that b-EWSA is NP-hard already when $b \geq 2$. Notice that there is a relatively straightforward reduction from 3-Partition to b-EWSA if $b \geq 3$. To show the result for any $b \geq 2$, we use a different problem, Numerical Three-dimensional Matching (N3DM) (Garey & Johnson, 1979).

Theorem 2. b-EWSA *is NP-hard for any constant $b \geq 2$ even if the sharing graph is a clique.*

Proof. We present a polynomial-time many-one reduction from the strongly NP-hard Numerical Three-dimensional Matching (N3DM) problem (Garey & Johnson, 1979). Therein, given 3 multisets of positive integers X, Y, Z, each containing m elements, and a bound T, the task is to decide whether there is a partition $S_1, S_2, \ldots, S_m$ of $X \cup Y \cup Z$ such that each S_i contains exactly one element from each of X, Y, Z and the sum of numbers in each S_i is equal to T.

Given an instance (X, Y, Z) of N3DM, we construct an instance of b-EWSA as follows. Without loss of generality, assume all elements from $X \cup Y \cup Z$ are smaller than T and the sum of them is equal to $B \coloneqq mT$. We set the goal $k \coloneqq (B^2 + B + 1)T$. We create 3 groups of agents corresponding to the 3 multisets X, Y, Z. For each $x_i \in X$, we create an agent a_i^1 in group 1 who holds a *large* resource that is valued as k by agent a_i^1 and $B^2T + x_i$ by all other agents. For each $y_i \in Y$, we create an agent a_i^2 in group 2 who holds a *middle* resource that is valued as k by agent a_i^2 and $BT + y_i$ by all other agents. For each $z_i \in Z$, we create an agent a_i^3 in group 3 who holds a *small* resource that is valued as z_i by agent a_i^3 and 0 by all other agents. We let the input sharing graph $\mathcal{G}_\mathrm{s}$ be a clique over the agents.

In the initial allocation, all $2m$ agents in groups 1 and 2 have utility exactly k and all m agents from group 3 have utility smaller than k.

$\Rightarrow$: If (X, Y, Z) is a "yes"-instance of N3DM, then using partition $S_1, S_2, \ldots, S_m$ with the claimed properties, we can find a sharing δ such that each agent participates in at most two sharings and has utility at least k under Π^δ as follows. For each $S_i = \{x_{i_1}, y_{i_2}, z_{i_3}\}$ with $x_{i_1} + y_{i_2} + z_{i_3} = T$, we let agent $a^1_{i_1}$ share its big resource with $a^3_{i_3}$ and let agent $a^2_{i_2}$ share its middle resource with $a^3_{i_3}$ so that agent $a^3_{i_3}$ has utility $B^2T + x_{i_1} + BT + y_{i_2} + z_{i_3} = (B^2 + B + 1)T = k$. Since each S_i contains exactly one element from each of X, Y, Z, we have that each agent in group 3 has value exactly k under Π^δ. Thus all agents have value at least k under Π^δ. Moreover, every agent in groups 1 and 2 participates in exactly one sharing and every agent in group 3 participates in exactly two sharings.

$\Leftarrow$: If the instance of b-EWSA is a "yes"-instance, then there is a sharing δ such that every agent has utility at least k under Π^δ. For each agent a^3_i in group 3, let $S'_i = \Pi^\delta(a^3_i)$ be the set of resources to which a^3_i has access under Π^δ. Since each agent a^3_i in group 3 values all small resources held by other agents as 0, without loss of generality, we can assume S'_i contains no small resources except for the one initially held by a^3_i itself. Notice that from the viewpoints of agents in group 3, the sum of value of all middle resources is smaller than that of one big resource. Since agent a^3_i has utility at least k under Π^δ, S'_i contains at least one big resource. It follows that each S'_i contains exactly one big resource as each big resource can be shared by at most one agent from group 3 and there are m agents in group 3 and m big resources. Next, to guarantee that each agent a^3_i has utility at least k, each S'_i should contain at least one middle resource. Again, since each middle resource can be shared by at most one agent and there are m agents in group 3 and m middle resources, each S'_i contains exactly one middle resource. Thus, each S'_i contains exactly one big resource, one middle resource, and one small resource; and each resource is contained in exactly one S'_i. According to the construction, based on $S'_1, S'_2, \ldots, S'_m$, we can find a partition $S_1, S_2, \ldots, S_m$ of $X \cup Y \cup Z$ such that each S_i contains exactly one element from each of X, Y, Z and the sum of the numbers in each S_i is at least $k - B^2T - BT = T$. Since the sum of all elements from $X \cup Y \cup Z$ is mT, the sum of the numbers in each S_i is exactly T. Therefore, (X, Y, Z) is a "yes"-instance of N3DM. □

5. Reducing Envy by Sharing

In this section, we study the problem of reducing envy through sharing. This scenario comes in constrast to the scenarios regarding the social welfare concepts from the previous section. The social welfare concepts are global measures that do not employ comparisons between agents. However, the concept of envy is based on agent-to-agent comparisons. As a result, considering reducing envy, in addition to the sharing graph, we use the attention graph describing the possible comparisons.

Definition 7. *Given an initial complete allocation π of m resources $\mathcal{R}$ to n agents $\mathcal{A}$, a sharing graph $\mathcal{G}_s$, an attention graph $\mathcal{G}_t$, and a non-negative integer k, Envy Reducing Sharing Allocation (ERSA) asks if there is a* simple 2-sharing *δ such that the number of envious agents $|\operatorname{Env}(\Pi^\delta)| \leq k$.*

u	r_j^{a}	$\mathring{r}_j^{\mathrm{a}}$	r_j^{p}	r^s
a_i	$[\{v_i, v_j\} \in E]$	0	3	3
p_i	1	1	3	3
s	0	0	0	3

Table 2: Utility functions in the proof of Theorem 3. We use the Iverson bracket notation: for some logical expression X, $[X]$ is one if X is `true` and zero otherwise.

In the above definition we restrict that the sharing is a *simple* 2*-sharing*, that is, 1-bounded 2-sharing. Indeed, this quite restricted problem variant for reducing envy in this setting already turns out to be NP-hard. In fact, this holds even in a special case when the attention graph and the sharing graph are (bidirectional) cliques and the goal is to decrease the number of envious agents by one, as shown in Theorem 3.

Theorem 3. ERSA *is* NP*-hard even if the attention graph and the sharing graph are (bidirectional) cliques, the goal is to reduce the number of envious agents by at least one, and there are three different utility values.*

Proof. We present a polynomial-time many-one reduction from the NP-hard Independent Set problem. In this problem, for an undirected graph $G = (V, E)$ and an integer ℓ, we ask whether G contains a subset of at least ℓ vertices that are pairwise non-adjacent.

Let $I = (G, \ell)$ be an instance of Independent Set with $G = (V, E)$ and $V = \{v_1, v_2, \ldots, v_n\}$. We construct an instance of ERSA as follows. For each vertex $v_i \in V$, we create an agent a_i who initially has two resources r_i^{a} and $\mathring{r}_i^{\mathrm{a}}$. Moreover, we add ℓ more agents: *providers* p_1, p_2, $\ldots$, $p_{\ell-1}$ and a *special provider* s. Initially, we allocate a resource r_i^{p} to each provider p_i and a resource r^s to the special provider. Next, we specify utility functions (see also Table 2). Each agent a_i has value 1 for each resource r_j^{a} for which $\{v_i, v_j\} \in E$, $j \in [n]$, and value 0 for the remaining resources r_j^{a} (in particular, agent a_i gives value 0 to its own resource). Furthermore, each agent a_i has value 0 for all $\mathring{r}_j^{\mathrm{a}}$, $j \in [n]$, and value 3 for the remaining resources. Regarding the provider p_i, it gives value 1 to resources in $\{r_1^{\mathrm{a}}, r_2^{\mathrm{a}}, \ldots, r_n^{\mathrm{a}}\} \cup \{\mathring{r}_1^{\mathrm{a}}, \mathring{r}_2^{\mathrm{a}}, \ldots, \mathring{r}_n^{\mathrm{a}}\}$ and value 3 to all other resources. The special provider s has value 3 for its own resource r^s and value 0 for all others. The attention graph is a (bidirectional) clique and the sharing graph is also a clique, so every two agents can share their resources. By the construction, initially there are n envious agents: $\{a_1, a_2, \ldots, a_n\}$. Thus, we conclude the construction by setting the target number $k := n-1$, so we aim at decreasing the number of envious agents by at least 1.

In what follows, we show that there is an independent set of size at least ℓ in G if and only if there is a simple 2-sharing δ such that $|\operatorname{Env}(\Pi^\delta)| \leq k$, that is, the number of envious agents is at least by one smaller than before the sharing. Before we start proving this claim, we emphasize that in every simple 2-sharing the special provider s is not envious.

$\Rightarrow$: Suppose that there is an independent set S of size ℓ in G. We denote by $S' \subseteq \{a_1, a_2, \ldots, a_n\}$ the set of ℓ agents corresponding to the vertices in S. Then, each agent in S' can share with a different provider (including the special provider) and increase its own value to 3. After this sharing, denoted by δ, agents in S' do not envy any provider

since from their point of view each provider has a bundle of value 3. In addition, since vertices in S are pairwise non-adjacent, agents in S' do not envy each other as they see each other having a bundle of value $3+0+0=3$. Finally, agents in S' do not envy agents in $\{a_1, a_2, \ldots, a_n\} \setminus S'$, whose bundles have a value of at most 1. Hence, as required by the claim, we have $|\operatorname{Env}(\Pi^\delta)| \leq (\ell+n) - \ell - 1 = k$ envious agents.

$\Leftarrow$: Suppose that there is a simple 2-sharing δ such that $|\operatorname{Env}(\Pi^\delta)| \leq k$. Let N be the set of non-envious agents after the sharing; for the reasons of the argument's clarity, we exclude the special provider s from N. Since overall there are $n+\ell$ agents and at most k of them are envious, we have $|N| \geq (n+\ell) - k - 1 = \ell$. Denote $N_\mathrm{a} = N \cap \{a_1, a_2, \ldots, a_n\}$ and $N_p = N \cap \{p_1, p_2, \ldots, p_{\ell-1}\}$. Then $|N_\mathrm{a}| + |N_\mathrm{p}| = |N| \geq \ell$. Since $|N_\mathrm{p}| \leq \ell - 1$, we have $N_\mathrm{a} \neq \emptyset$. We show that actually $N_\mathrm{p} = \emptyset$. Suppose towards a contradiction that $N_\mathrm{p} \neq \emptyset$ and consider some agents $a^* \in N_\mathrm{a}$ and $p^* \in N_\mathrm{p}$. Initially, $u_{a^*}(\pi(a^*)) = 0$ and $u_{a^*}(\pi(p^*)) = 3$. Since a^* is not envious after sharing δ, a^* must get one resource as an effect of sharing with some provider. Consequently, $u_{p^*}(\Pi^\delta(a^*)) = 1+1+3 = 5$. Since initially $u_{p^*}(\pi(p^*)) = 3$ and p^* is not envious after sharing δ, p^* needs a resource from another provider, as p^* requires a resource of value at least 2 to overcome this envy. However, this would again leave a^* envious because a^* cannot participate in more than one sharing. Thus, $N_\mathrm{p} = \emptyset$, and hence $N = N_\mathrm{a} \subseteq \{a_1, a_2, \ldots, a_n\}$.

Since agents in N are non-envious, it requires that all of them share with a unique provider. For every two distinct agents $a_i, a_j \in N$, let $r^\mathrm{p}_{i'}$ and $r^\mathrm{p}_{j'}$ be the resources shared respectively to a_i and a_j. Due to the fact that a_i is not envious towards a_j, we have that $u_{a_i}(\Pi^\delta(a_i)) \geq u_{a_i}(\Pi^\delta(a_j))$, where $\Pi^\delta(a_i) = \{r^\mathrm{a}_i, \mathring{r}^\mathrm{a}_i, r^p_{i'}\}$ and $\Pi^\delta(a_j) = \{r^\mathrm{a}_j, \mathring{r}^\mathrm{a}_j, r^p_{j'}\}$. Since $u_{a_i}(r^\mathrm{a}_i) = u_{a_i}(\mathring{r}^\mathrm{a}_i) = u_{a_i}(\mathring{r}^\mathrm{a}_j) = 0$ and $u_{a_i}(r^\mathrm{p}_{i'}) = u_{a_i}(r^\mathrm{p}_{j'}) = 3$, we get $u_{a_i}(r^\mathrm{a}_j) = 0$, which means that $\{v_i, v_j\} \notin E$. Thus the corresponding vertices for agents in N form an independent set of size at least ℓ in G. □

Theorem 3 in fact constitutes a strong intractability result and it calls for further studies on other specific features of the input. We counteract the intractability result of ERSA (Theorem 3) by considering cases with few agents, tree-like graphs, identical utility functions, or few resources.

5.1 Reducing Envy for Few Agents

We start by considering the case with few agents, where our main result is that ERSA is fixed parameter tractable with respect to the number of agents. Before presenting the FPT algorithm, we observe that there exist straightforward brute-force algorithms that can solve ERSA in polynomial time if the number of agents or the number of resources is constant.

Observation 1. *There is an algorithm solving the* ERSA *problem with n agents and m resources in $O\left((m+1)^{n+1} \cdot n\right)$ time and another one that solves the same problem in $O\left(n^{m+1} \cdot m\right)$ time.*

Proof. We enumerate all possible sharings and compute for each sharing the number of envious agents. When the number n of agents is constant, for each agent a we guess a resource $r \in \mathcal{R} \cup \{\diamond\}$ with which it might be involved in the sharing, where $\diamond$ is a dummy resource used to model the case that the agent is not involved in a sharing. Then for each

case we check whether the guess induces a valid simple 2-sharing: If some agent a owns the resource r we guessed for a and we have guessed the same resource for exactly one neighbor a' of a, then this means agent a shares r with a'. For each valid simple 2-sharing, we compute the number of envious agents. Since there are $(m+1)^n$ cases and for each case we can check the validity and compute the number of envious agents in $O(nm)$ time, the whole algorithm runs in $O\left((m+1)^{n+1} \cdot n\right)$ time.

Consider another case, in which the number of resources m is constant. Then, for each resource r and its respective agent a such that $r \in \pi(a)$, we guess one agent a' who participates in sharing r by a. If $a = a'$, then we assume that r is not shared. Next, we check whether the guess induces a valid simple 2-sharing, that is, whether each agent is involved in at most one sharing. For each valid simple 2-sharing, we compute the number of envious agents. Since there are n^m cases and for each case we can check the validity and compute the number of envious agents in $O(nm)$ time, the whole algorithm works in $O\left(n^{m+1} \cdot m\right)$ time. □

From the negative pespective, factor m (resp. n) appears in the base of the exponent in the running-time formula from Observation 1. Hence, the running times of such an algorithm skyrockets with large m (resp. n), even for small, fixed values of n (resp. m). We improve this in Theorem 4 by showing that ERSA is fixed-parameter tractable with respect to the number of agents n. In contrast, we will show in Theorem 8 that this is not the case for the number of resources m even in a very restricted case.

Theorem 4. *There is an algorithm solving the* ERSA *problem with n agents and m resources in $O\left((2n)^n \cdot m^2\right)$ time.*

The high-level idea behind Theorem 4 is as follows. In order to find a desired sharing, our algorithm guesses *target agents*—a set of at least $n - k$ agents that do not envy in the desired sharing—and a *sharing configuration*—a set of ordered pairs of agents that share some resource in the desired sharing. Then, for such a guessed pair, the algorithm tests whether the desired sharing indeed exists. If it is true for at least one guessed pair, then the algorithm returns "yes"; otherwise, it returns "no." The main difficulty in checking the existence of the desired sharing is that we need to maintain the envy-freeness within target agents while increasing their utilities.

Before stating the algorithm more formally, we give some notation and definitions. Let C be a fixed set of target agents.

Definition 8. *A* sharing configuration *M for a set C of target agents is a set of arcs such that*

1. *M is a set of vertex-disjoint arcs and*
2. *if $(i,j) \in M$, then $\{i,j\} \in \mathcal{E}_s$ and $j \in C$.*

A simple 2-sharing δ is called a realization *of M if δ only specifies the shared resource for each arc in M; formally, for each $(i,j) \in M$, we have that $\big(\delta(\{i,j\}) \neq \emptyset\big) \wedge \big(\delta(\{i,j\}) \in \pi(i)\big)$, and for each $\{i,j\}$ with $\delta(\{i,j\}) \neq \emptyset$, we have that $\big((i,j) \in M \wedge \delta(\{i,j\}) \in \pi(i)\big) \vee \big((j,i) \in M \wedge \delta(\{i,j\}) \in \pi(j)\big)$. A realization δ is* feasible *if $C \cap \mathrm{Env}(\Pi^\delta) = \emptyset$. i.e., no agent in C will be envious under δ.*

Note that a sharing configuration does not only describe shares of a proper simple 2-sharing but also ensures that the shared resources are indeed shared "to" the target agents; we justify this restriction later in Lemma 2.

Let us fix a sharing configuration M for C. For each target agent a_i, we define a set P_i^0 of *initially possible resources* that a_i might get in some realization of M. For convenience, we augment each P_i^0 with a dummy resource d_i that has utility zero for every agent. Formally, we have

$$P_i^0 \coloneqq \begin{cases} \pi(j) \cup \{d_i\} & \text{if } \exists j \text{ such that } (j,i) \in M, \\ \{d_i\} & \text{otherwise.} \end{cases} \tag{1}$$

Note that for each agent a_i, there is at most one j such that $(j,i) \in M$ since the edges in M are vertex-disjoint by definition.

For each target agent $a_i \in C$, we define a *utility threshold* t_i—the smallest utility agent a_i must have such that a_i will not envy agents *outside* C. Formally, if there is at least one agent $a_j \notin C$ such that $(a_i, a_j) \in \mathcal{G}_\mathrm{t}$, then

$$t_i \coloneqq \max_{a_j \notin C, (a_i,a_j) \in \mathcal{G}_\mathrm{t}} u_i(\pi(j)), \tag{2}$$

otherwise, $t_i \coloneqq 0$. If some target agent cannot achieve its utility threshold by obtaining at most one of its initially possible resources, then there is no realization of M such that the agent does not envy. We express it more formally in Observation 2.

Observation 2. *There is no feasible realization of M in which some agent $a_i \in C$ gets a resource $r \in P_i^0$ such that $u_i(\pi(i) \cup \{r\}) < t_i$.*

For each target agent $a_i \in C$ and a family of sets of possible resources for the target agents during the execution of our algorithm, we define a set of *forbidden resources*.

Definition 9. *Let $C = \{a_1, a_2, \ldots, a_q\}$ and let $\mathcal{P} = \{P_1, P_2, \ldots, P_q\}$ be a family of sets of possible resources for the target agents. Then resource $r \in P_i$ is a* forbidden resource *for some target agent a_i if there is some target agent a_j with $(a_j, a_i) \in \mathcal{G}_t$ such that*

$$\max\{u_j(\pi(j) \cup \{r'\}) \mid r' \in P_j\} < u_j(\pi(i) \cup \{r\}),$$

that is, if agent a_i gets resource r, then agent a_j will envy a_i even if a_j gets its most valuable resource from P_j. We denote the set of all forbidden resources for a_i as $F_i(\mathcal{P})$.

Observe that in every feasible realization no target agent gets one of its forbidden resources since otherwise there is another target agent that envies.

Observation 3. *Let $\mathcal{P}$ be a family of possible resources for the target agents. In every feasible realization no target agent a_i gets a resource from $F_i(\mathcal{P})$.*

Based on the above observations, Algorithm 1 tests whether for a pair of a set C of target agents and a sharing configuration M there is a feasible realization. The algorithm keeps track of the possible resources P_i for each target agent a_i. Starting with each P_i equal to the corresponding set of initially possible resources, it utilizes Observation 2 and removes the "low-utility" resources. Then, utilizing Observation 3, the algorithm finds all

DoesFeasibleRealizationExist$(\mathcal{G}_\mathrm{t}, \pi, \{u_i\}_{i\in C}, C, M)$
for *each agent* $a_i \in C$ **do**
  $P_i \leftarrow P_i^0 \setminus \{r \in P_i \mid u_i(\pi(i) \cup \{r\}) < t_i\}$;
end
repeat
  $B \leftarrow \bigcup_{a_i \in C} F_i(\{P_1, P_2, \ldots, P_{|C|}\})$;
  $P_i \leftarrow P_i \setminus B$ for each agent $a_i \in C$;
until $B = \emptyset$;
if $\exists i$ *with* $P_i = \emptyset$ **then return** *"no"* **else return** "yes";

Algorithm 1: Testing existence of a feasible realization of sharing configuration M for set C of target agents.

forbidden resources for a particular collection of the possible resources for the target agents and eliminates the forbidden resources. This procedure is repeated exhaustively. Finally, if at least one of the possible resource sets is empty, the algorithm outputs "no." Otherwise, the algorithm returns "yes" since at least one resource remained in the set of possible resources for every target agent.

After applying Observation 2 and 3 to Algorithm 1 and proving its correctness (Lemma 1 and Lemma 2), we finish the proof of Theorem 4.

Lemma 1. *In Algorithm 1, if some P_i is empty after the repeat-loop, then there is no feasible realization for M for C.*

Proof. Suppose towards a contradiction that P_i is empty after the repeat-loop and there is a feasible realization δ for M such that $C \cap \mathrm{Env}(\Pi^\delta) = \emptyset$. Let $r^* \in \Pi^\delta(i) \setminus \pi(i)$ be the resource shared to agent i. For any agent $j \in C$ with $(j, i) \in \mathcal{E}_\mathrm{t}$, let $r_j \in \Pi^\delta(j) \setminus \pi(j)$ be the resource shared to agent j. We remark that here resources r^* and r_j could be dummy resources. Since $j \in C$ and $C \cap \mathrm{Env}(\Pi^\delta) = \emptyset$, we have $j \notin \mathrm{Env}(\Pi^\delta)$, and hence j does not envy i after δ. Thus $u_j(\pi(j)) + u_j(r_j) \geq u_j(\pi(i)) + u_j(r^*)$. Since this holds for any agent $j \in C$ with $(j, i) \in \mathcal{E}_\mathrm{t}$, resource $r^* \in S_i$ is not a forbidden resource, which contradicts that S_i becomes empty after deleting blocking resources. □

Lemma 2. *If there is a simple 2-sharing σ such that $C \cap \mathrm{Env}(\Pi^\sigma) = \emptyset$, then there is a sharing configuration M for C that has a feasible realization and Algorithm 1 outputs "yes"; otherwise, the algorithm outputs "no".*

Proof. We first show that if there is a simple 2-sharing σ such that $C \cap \mathrm{Env}(\Pi^\sigma) = \emptyset$, then there is a sharing configuration M for C that has a feasible realization δ. Recall that all sharing configurations have the restriction that the shared resources are indeed shared "to" the target agents. We now justify this restriction. Let $E = \{\{i, j\} \in \mathcal{E}_\mathrm{s} \mid \sigma(\{i, j\}) \neq \emptyset\}$ be the set of edges where there is a sharing in σ. Let $E_1 \subseteq E$ be the set of edges that could be used in a sharing configuration for C, i.e.,

$$E_1 = \{\{i, j\} \in E \mid \big(\sigma(\{i, j\}) \in \pi(i) \wedge j \in C\big) \vee \big(\sigma(\{i, j\}) \in \pi(j) \wedge i \in C\big)\}.$$

We construct a new simple 2-sharing δ by restricting δ on E_1:

$$\delta(\{i,j\}) = \begin{cases} \sigma(\{i,j\}) & \text{if } \{i,j\} \in E_1, \\ \emptyset & \text{otherwise.} \end{cases}$$

Then we define the sharing configuration as

$$M = \{(i,j) \mid \delta(\{i,j\}) \neq \emptyset \wedge \delta(\{i,j\}) \in \pi(i)\}.$$

Now it is clear that M is a sharing configuration for C and δ is a realization of M. Moreover, since all sharings through edges in $E \setminus E_1$ only increase the bundles of agents in $\mathcal{A} \setminus C$, we have that

$$\begin{cases} u_i(\Pi^\delta) = u_i(\Pi^\sigma) & \forall i \in C, \\ u_i(\Pi^\delta) \leq u_i(\Pi^\sigma) & \text{otherwise.} \end{cases}$$

Since $C \cap \mathrm{Env}(\Pi^\sigma) = \emptyset$, we have that $C \cap \mathrm{Env}(\Pi^\delta) = \emptyset$. Therefore, δ is a feasible realization.

Next, we show that there is a sharing configuration M for C that has a feasible realization if and only if Algorithm 1 outputs "yes". According to Lemma 1, if the algorithm outputs "no", then there is no feasible realization for M for C. On the other hand, if the algorithm outputs "yes", then at the end of the algorithm we get $P_i \neq \emptyset$ for each agent $a_i \in C$. We can build a simple 2-sharing δ by choosing the most valuable resource from P_i for each agent $a_i \in C$. Notice that P_i are disjoint, so there is no possibility that two agents would receive the same resource. If $P_i = \{d_i\}$, then no resource is shared to agent a_i. It is clear that δ is a realization for M for C. Moreover, for any agent $a_i \in C$, a_i does not envy agents outside C (as resources that do not have high enough utility for a_i such that a_i does not envy agents outside C are removed from P_i and $P_i \neq \emptyset$ at the end) and a_i does not envy agents in C (as $B = \emptyset$). Therefore, δ is a feasible realization for M for C. □

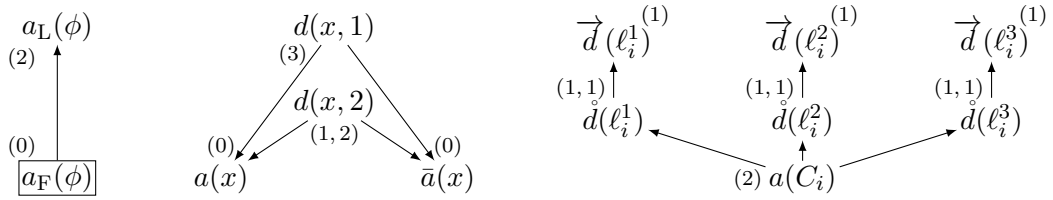
Eventually, we are set to present the proof of Theorem 4.

Proof of Theorem 4. According to Lemma 2, to solve an instance of ERSA, it is enough to test whether there is a pair of a target subset and a sharing configuration that has a feasible realization. Since, checking a feasible realization, due to Lemma 2, can be done by Algorithm 1, we check all such possible pairs and return "yes" if there is (at least) one that has a feasible realization.

There are $O(2^n)$ possible target sets and at most n^n possible sharing configurations per target set, which gives $O\left((2n)^n\right)$ cases. For each case, we apply Algorithm 1. Therein, the for-loop takes $O(nm)$ time. Concerning the repeat-loop that runs at most m times, computing the set B takes $O(nm)$ time; thus, the repeat-loop takes $O(nm^2)$ time. Finally, Algorithm 1 runs in $O(nm^2)$ time and ERSA can be solved in $O\left((2n)^n \cdot m^2\right)$ time. □

Next, we show that restricting the parameter "number k of envious agents" does not help to make ERSA solvable in polynomial time. Indeed, already with $k = 1$, our problem remains computationally hard.

Theorem 5. ERSA *is* NP*-hard even if the goal is to reduce the number of envious agents from one to zero and the unanimous utility functions take four different values.*



(a) The leader gadget for formula ϕ.

(b) The variable gadget for variable x.

(c) The clause gadget for clause $C_i = (\ell_i^1 \vee \ell_i^2 \vee \ell_i^3)$.

Figure 2: The gadgets in the construction in the proof of Theorem 5. The utility function is unanimous. The envious agents are framed. Numbers in brackets indicate an initial allocation (e.g., (0) denotes an empty bundle; $(1, 2)$ means a bundle of a value-one and a value-two resource).

Proof. We provide a polynomial-time many-one reduction from the NP-hard 3-CNF-Sat problem that asks whether a given Boolean expression in conjunctive normal form with each clause of size at most three is satisfiable. From now on we stick to the following notation. Let $\phi = \bigwedge_{i \in [\bar{m}]} C_i$ be a 3-CNF formula over a set $X = \{x_1, x_2, \ldots, x_{\bar{n}}\}$ of variables where a clause C_i, $i \in [\bar{m}]$, is of the form $(\ell_i^1 \vee \ell_i^2 \vee \ell_i^3)$. We use a standard naming scheme and call each $\ell_i^{\cdot}$ a literal.

We first describe the *leader gadget*, the *clause gadget*, and the *variable gadget* (see Figure 2 for an overview of the construction). Then, we show how to connect our gadgets to achieve a desired instance of ERSA. Eventually, we prove the reduction's correctness preceding it by a discussion on the structure of the built instance. Notably, in our construction we use a unanimous utility function, that is, each agent we introduce has the same utility function over all introduced resources.

Construction The leader gadget consists of two agents: the *leader* $a_{\mathrm{L}}(\phi)$ and the *follower* $a_{\mathrm{F}}(\phi)$. The follower is not assigned any resource, while the leader has a resource of value two. There is a directed arc from the follower to the leader. Thus, initially, the follower envies the leader.

The variable gadget for a variable $x \in X$ consists of two *value agents* $a(x)$ and $\bar{a}(x)$, and two *dummy agents* $d(x, 1)$ and $d(x, 2)$. The two value agents represent, respectively, assigning `true` and `false` to x. Both dummy agents are paying attention to both value agents (but not vice versa). Initially, three resources are allocated. The resource of value three is allocated to $d(x, 1)$ and two resources, one of value one and one of value two, are allocated to $d(x, 2)$. As a result, initially, no agent building the gadget envies.

To describe the clause gadget, let us fix a clause $C_i = (\ell_i^1 \wedge \ell_i^2 \wedge \ell_i^3)$. For each literal ℓ in the clause we add two agents: the *donor* $\overrightarrow{d}(\ell)$ and the *recipient* $\mathring{d}(\ell)$. The donor initially has a single one-valued resource, while the recipient initially gets two one-valued resources. As for the attention relation, for each literal ℓ, there is an arc $(\mathring{d}(\ell), \overrightarrow{d}(\ell))$, that is, the arc points from the recipient to the donor. Eventually, the gadget contains the *root* agent $a(C_i)$ that gets a single resource of value two. The root agent pays attention to recipients $\mathring{d}(\ell_i^1)$, $\mathring{d}(\ell_i^2)$, and $\mathring{d}(\ell_i^3)$. Observe that, initially, no agent within this gadget envies.

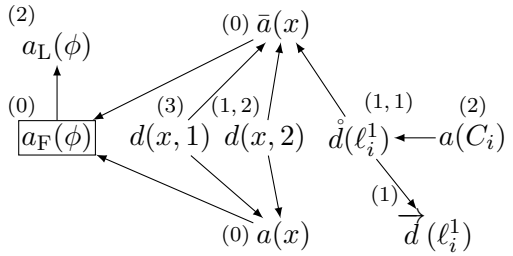


Figure 3: Connections of gadgets for literal ℓ_i^1 (over variable x) of some clause C_i. Envious agents are framed. Other possible literals of C_i are omitted for clarity.

We obtain the full construction by interconnecting the gadgets (see Figure 3). Consider a single copy of the leader gadget, one variable gadget per variable, and one clause gadget per clause. First, we connect every value agent with an arc directed from the value agent to the follower. Note that we do not introduce any envy because the follower has no resources initially. Finally, for each literal ℓ_i^j, $j \in \{1, 2, 3\}$, $i \in [\bar{m}]$, we add an arc from a respective recipient $\mathring{d}(\ell_i^j)$ to the value agent of the corresponding variable agent; for example, for $\ell_i^j = \neg x_y$, $y \in [\bar{n}]$, we would add an arc $(\mathring{d}(\ell_i^j), \bar{a}(x_y))$. The sharing graph is just the same as the underlying undirected graph of the attention graph.

The constructed attention graph, the sharing graph, the introduced agents, the resources allocated by the initial allocation, and their utilities together with the desired number $k \coloneqq 0$ of envious agents form an instance of ERSA, which is clearly computable in polynomial time.

Correctness We show that ϕ is satisfiable if and only if there exists a simple 2-sharing for the above instance such that no agent is envious.

$\Rightarrow$: Suppose there exists a satisfiable truth assignment $\mathbf{x}$ of ϕ. We first make the leader share with the follower, making the follower non-envious. Next, in each variable gadget, for every variable x set to `true`, we let agent $a(x)$ get the two-value resource and $\bar{a}(x)$ get the three-value resource through sharing with the corresponding dummy agents; otherwise, we let agent $a(x)$ get the three-value resource and $\bar{a}(x)$ get the two-value resource. Finally, for each clause C, since $\mathbf{x}$ is a satisfiable truth assignment, there must be a literal $\ell \in C$ that is assigned `true`, which means that the corresponding value agent gets a two-value resource. Hence in the clause gadget of C, recipient $\mathring{d}(\ell)$ shares one of its one-value resource with the root agent $a(C)$. Thus, agent $a(C)$'s utility is effectively three so two other two recipients share with their donors to also obtain the utility of three. It is easy to verify that no agent is envious under this simple 2-sharing.

$\Leftarrow$: Suppose there exists a simple 2-sharing for the above instance such that no agent envies. Since none of the value agents has any resource, the only way to make the follower non-envious is to share the leader's resource. This makes all value agents envious because they look at the follower that has obtained a resource of value two. Hence, to decrease the number of envious agents to zero, one has to provide each value agent with a resource of value at least two. Since respective recipients that are connected to the value agents have only one-valued resources, value agents have to share the resources of the dummy agents.

Among these only two suitable resources exist, two- and three-valued ones. Thus, in each variable gadget, one of the value agents needs to get a resource of value two and the other a resource of value three.
We construct a truth assignment by setting a variable x to `true` if and only if agent $a(x)$ gets a two-value resource. Next we show that this truth assignment satisfies every clause of ϕ. Towards a contradiction, suppose there exists an unsatisfied clause C in the constructed truth assignment. Then by construction, all the three recipient agents for this clause look at a value agent with a three-valued resource. Since no agent envies, each of these three recipient agents has to share with one neighbor to increase its utility to three. They cannot share with the root agent as otherwise the root agent will be envious. Clearly, this envy cannot be avoided since the root agent cannot participate in any further sharing. As a result, the three recipient agents share with the three donor agents. This, however, also makes the root agent envious. Again, one cannot avoid this envy by sharing because there is no other agent that the root agent could share with. Hence, we arrive at the contradiction. □

5.2 Reducing Envy for Tree-like Graphs

We proceed by studying how the tree-like structure of the sharing graph and the attention graph influences the computational complexity of ERSA. Studying tree-likeness, we hope for tractability for quasi-hierarchical social networks, where agents at the same level of the hierarchy influence each other but they rather do not do so in a cross-hierarchical manner.

Theorem 3 shows that when both graphs are (bidirectional) cliques ERSA is NP-hard. We continue to focus on the case when the underlying graph of the attention graph is the same as the sharing graph. Note that this restriction appears naturally when assuming that one may envy everybody one knows and one may share only with known people. On the one hand, Theorem 6 shows that if the sharing graph is a path, a tree or being very close to a tree (corresponding to a "hierarchical network"), then we can solve ERSA in polynomial time. On the other hand, Theorem 6 establishes that for sharing graphs being "far from a path," presumably there is no algorithm whose exponential growth in the running time depends only on the "distance from path."

Theorem 6. *When the underlying graph of the attention graph is the same as the sharing graph,* ERSA *can be solved in* $O\left(\mathrm{tw} \cdot n \cdot (8m)^{\mathrm{tw}+2}\right)$ *time if the sharing graph has a constant*[3] *treewidth* tw*, and* ERSA *is* W[1]*-hard with respect to the pathwidth of the sharing graph even if there are four different utility functions.*

Proof of the first part. We give a polynomial-time algorithm based on dynamic programming based on a (nice) tree decomposition. Let us fix some instance of ERSA with agents $\mathcal{A}$, resources $\mathcal{R}$, and a sharing graph $\mathcal{G}_{\mathrm{s}}$. Let G with treewidth tw be the underlying graph of $\mathcal{G}_{\mathrm{s}}$. By $\mathcal{T} = (T, \{X_1, X_2, \ldots, X_{|V(T)|}\})$ we denote a nice tree decomposition of G; here T is a tree, $V(T)$ is the set of vertices of T called *nodes*, and X_i are the decomposition bags. Since we assume constant treewidth, we can precompute $\mathcal{T}$ in polynomial time (Cygan et al., 2015). Without loss of generality, we assume that $\mathcal{T}$ is given. Before presenting the algorithm, we first define further concepts and notation.

3. We make this assumption so that we can assume that a nice tree decomposition is given as input since it can be computed in $f(\mathrm{tw}) \cdot n$ time, as mentioned in Section 2.3.

A *bundle configuration* $b\colon X_t \to 2^{\mathcal{R}}$ *for bag* X_t is a function which maps every agent in X_t to a bundle of resources. We say a sharing δ *realizes* bundle configuration b if $\Pi^{\delta}(a) = b(a)$ for each $a \in X_t$. Bundle configuration b is *feasible* if there exists a simple 2-sharing δ which realizes b. Note that π is a trivial bundle configuration that can be realized by an empty sharing. Let us fix a node $t \in T$. By B_t we denote the set of all feasible bundle configurations for bag X_t, and by V_t we denote the union of all the bags in the subtree of T rooted in t (including X_t itself). Furthermore, we write $\mathrm{Env}_{[V_t]}(\delta)$ for the set of all envious agents in the subinstance induced by the agents in V_t under some sharing δ. Note that $\mathrm{Env}_{[V_t]}(\delta)$ does not contain an agent a if a only envies agents outside of V_t under δ. For some bundle configuration b for bag X_t and a bag $X_{t'} \subseteq X_t$, we denote by $b[X_{t'}]$ the bundle configuration described by b restricted to $X_{t'}$.

Consider a bag X_t of the tree decomposition, a feasible bundle configuration $b \in B_t$, and a subset $S \subseteq X_t$ of the agents of bag X_t. We define a function $f[t, b, S]$ which yields the minimum number of envious agents in the subinstance induced by V_t under a sharing δ such that δ realizes b and $\mathrm{Env}_{[V_t]}(\delta) \cap X_t = S$. Formally, for every bundle configuration $b \in B_t$ and every subset $S \subseteq X_t$, the definition of f reads as follows:

$$f[t, b, S] \coloneqq \min\{\mathrm{Env}_{[V_t]}(\delta) \mid (\delta \text{ realizes } b) \wedge (\mathrm{Env}_{[V_t]}(\delta) \cap X_t = S)\}.$$

For convenience, we assume $f[t, b, S] = +\infty$, if no sharing δ exists for a given collection of f's parameters. Taking $r \in V(T)$ as the root node of T, it is easy to see that our goal is to compute $f[r, \pi, \emptyset]$. Notice that since $\mathcal{T}$ is a nice tree decomposition, the root node X_t and all leaf nodes are empty sets.

Starting from the leaves of T, we compute the values of f bottom-up according to the following formulae regarding each of the four possible types of nice tree decomposition nodes: leaf node, introduce node, forget node, and join node.

Leaf node. Suppose t is a leaf node of T. Since (by definition of nice tree decompositions) $X_t = \emptyset$, then we have only one value $f[t, \pi, \emptyset] = 0$.

Introduce node. Suppose t is an introduce node with a child t' such that $X_t = X_{t'} \cup \{v\}$ for some $v \notin X_{t'}$. For each bundle configuration $b \in B_t$ and $S \subseteq X_t$, we first check whether b and S are compatible, that is, whether there is a sharing δ realizing b such that $\mathrm{Env}_{[V_t]}(\delta) \cap X_t = S$. To this end, we compute a set $E \subseteq X_t$ of agents who envy the agents in X_t assuming the bundles imposed by b. Observe that for each δ realizing b, it holds that $E \subseteq \mathrm{Env}_{[V_t]}(\delta) \cap X_t$. Thus, if $E \setminus S \neq \emptyset$, then b and S are incompatible, and we set $f[t, b, S] = +\infty$. By the property of tree decompositions, all neighbors of v that are in V_t are also in X_t. So if $v \notin E$, then v will not be envious in the subinstance induced by V_t under any sharing which realizes b. Thus, if $v \in S \setminus E$, then b and S are also incompatible, and we set $f[t, b, S] = +\infty$. Now for each $b \in B_t$ and $S \subseteq X_t$ with $E \subseteq S$ and $v \in E \Leftrightarrow v \in S$, we have that:

$$f[t, b, S] = \min\{|S \setminus S^*| + f[t', b[X_{t'}], S^*] \mid S \setminus E \subseteq S^* \subseteq S \cap X_{t'}\}.$$

Intuitively, the above formula says which choice of S^* is optimal for the given S and b with respect to the total number of envious agents. Since b guarantees that among the agents in X_t only agents in $E \subseteq S \subseteq X_t$ are envious, at least agents in $S \setminus E$ must have been already guaranteed being non-envious by sharings for the values of f for node t'. Thus we require that $S \setminus E \subseteq S^*$.

Forget node. Suppose t is a forget node with a child t' such that $X_t = X_{t'} \setminus \{w\}$ for some $w \in X_{t'}$. Then, for every $b \in B_t$ and $S \subseteq X_t$, we have that:

$$f[t, b, S] = \min\{f[t', b^*, S^*] \mid (b^*[X_t] = b) \wedge (S \subseteq S^* \subseteq S \cup \{w\})\}.$$

The intuition behind the above is as follows. Since we "forget" agent w and thus we will never consider it again, we want to find the optimal sharing among these ones that either make w envious or not. Hence, in the formula we seek the minimum over all possible values of f for t' that guarantee the same situation as the one described by b and S for all agents in $X_{t'}$ except for agent w, which we forgot in node t'.

Join node. Suppose t is a join node with children t_1 and t_2 such that $X_t = X_{t_1} = X_{t_2}$. Then, for each $b \in B_t$ and $S \subseteq X_t$, we have that:

$$f[t, b, S] = \min\{f[t_1, b, S_1] + f[t_2, b, S_2] - |S_1 \cap S_2| \mid S_1 \cup S_2 = S\}.$$

To see the correctness of the above formula it is crucial to recall that by the definition of tree decompositions it holds that $V_{t_1} \cap V_{t_2} = X_t$. Hence, we can simply add up the respective values of f for the children followed by subtracting $|S_1 \cap S_2|$ as the envious agents in $|S_1 \cap S_2|$ have been counted in both $f[t_1, b, S_1]$ and $f[t_2, b, S_2]$.

Finally, we proceed with the running time analysis of the algorithm. For each node t, we have $|B_t| \leq m^{\mathrm{tw}+1}$ since in every bundle configuration for X_t, each agent in X_t can get at most one additional resource compared to its initial bundle. Naturally, for each node t, we have at most $2^{\mathrm{tw}+1}$ subsets S_t. Since the nice tree decomposition has at most $O(\mathrm{tw} \cdot n)$ nodes (Cygan et al., 2015), it follows that we have at most $O(\mathrm{tw} \cdot n \cdot (2m)^{\mathrm{tw}+1})$ values of function f to compute. Values of an introduce node and a forget node can be computed, respectively, in $O(2^{\mathrm{tw}+1})$ (counting all possible subsets of $X_{t'}$ for S^*) and in $O(m)$ (there are at most this many bundle configurations b_t^*) time. Values of a join node can be computed by brute-forcing all possible pairs of S_1 and S_2 in $O(2^{2\mathrm{tw}+2})$ time. So the worst-case running time is $O\left(\mathrm{tw} \cdot n \cdot (8m)^{\mathrm{tw}+2}\right)$. □

Proof of the second part. We provide a parameterized reduction from the W[1]-hard Multicolored Clique problem (Downey & Fellows, 2013). Here, for an integer ℓ and an undirected graph $G = (V, E)$ in which each vertex is colored with one of ℓ colors, the goal is to find a set of ℓ pairwise adjacent, differently colored vertices. Without loss of generality, we assume that V can be partitioned into ℓ size-n sets $V_i = \{v_1^i, v_2^i, \ldots, v_n^i\}$, $i \in [\ell]$, where each V_i represents vertices of color i. Similarly, we assume that there are no edges between vertices of the same color. Subsequently, we describe how to construct an instance of ERSA given an instance (G, ℓ) of Multicolored Clique. Figure 4 illustrates the crucial gadgets of the reduction.

Before we present the construction of the instance of ERSA, we introduce some handy notation. Let us fix a pair of distinct colors i and j. We refer to the set of edges connecting the vertices of these colors as $E_{i,j} = \{\{v, v'\} \in E \mid v \in V_i, v' \in V_j\}$. Complementarily, let $\overline{E}_{i,j} = (V_i \times V_j) \setminus E_{i,j}$ and $\overline{E} = \bigcup_{i \neq j \in [\ell]} \overline{E}_{i,j}$. We say that an edge e is *forbidden* if $e \in \overline{E}$. Indeed, since forbidden edges are not part of G, they cannot appear between vertices of any clique in G.

Construction Our construction consists of the *vertex selection gadget* and the *certification gadget*. We describe them in the given order specifying how they relate to each other in the description of the certification gadget. For better understanding, we refer to Figure 4 showing the construction and Table 3 showing the utility functions.

For each color i, we build the *vertex selection* gadget consisting of three agents: a *selector* s_i, a *provider* p_i, and a *dummy* d_i. As for the attention relation, the provider attends the selector, that, in turn, attends the dummy. For each vertex $v \in V_i$ we create a *vertex resource* $r(v)$ and give all of them to the provider. The selector initially gets two resources $r(s_i, 1)$ and $r(s_i, 2)$. The same applies to the dummy that gets $r(d_i, 1)$ and $r(d_i, 2)$. The desired goal of the vertex selection gadget is that the provider shares with the selector exactly one vertex resource. We get this by making the selector initially envious of the dummy and ensuring that the dummy cannot share with the selector to remove the envy. To implement such a behavior, we set the utility function of the dummy and the provider to be 0 for all resources. The selector, however, gives utility 1 to both $r(d_i, 1)$ and $r(d_i, 2)$ and utility 2 to all vertex resources; the selector gives utility 0 to every other resource. Thus, under the initial allocation, the selector has utility 0 and envies the dummy, whose bundle has utility 2 for the selector.

For each forbidden edge, we build the *certification gadget*. Our goal is to represent a clique by the vertex resources that are shared with the selectors. Hence, the purpose of the certification gadget is to forbid selectors to obtain via a sharing the vertex resources representing the vertices that are not adjacent in G. To demonstrate the gadget, let us fix a pair $\{i, j\}$ of distinct colors such that $i < j$ and a forbidden edge $e = \{v, v'\} \in \overline{E}_{i,j}$. The certification gadget consists of two *certification* agents c_e^i and c_e^j, each having a resource—$r(c_e^i)$ and $r(c_e^j)$, respectively. Agent c_e^j attends c_e^i. We connect the certification gadget with the respective vertex selection gadgets by letting both certification agents attend the selectors of the respective colors, that is c_e^i attends s_i and c_e^j attends s_j. We make two certification agents being envious of the respective selectors according to the initial allocation. Further, we ensure that we can make one of the certification agents non-envious if and only if the respective selectors get vertex resources that do not represent forbidden edge e. Additionally, we make sure that sharing a resource from a selector to a certification agent cannot make the latter non-envious. To this end, c_e^i gives utility 1 to both resources initially given to s_i. Further, c_e^i gives utility 2 for vertex resource $r(v)$ and utility 1 for all other vertex resources of color i. We set analogously the utility function of c_e^j for the initial resources of s_j and the vertex resources of color j. Finally, both certification agents give each other's resource utility 3 and utility 0 for every other resource (including the ones they initially hold).

Finally we set $k := \binom{\ell}{2} n^2 - |E|$, which is the number of all forbidden edges. This finishes our construction of the instance of ERSA. Concerning the pathwidth, denote the set of all selectors, all providers and all dummies as B. There is a path decomposition $\mathcal{P} = (X_1, X_2, \ldots, X_m)$, where $m = |\overline{E}|$ is the number of forbidden edges, and each bag X_i contains all vertices in B and two certification agents for some forbidden edge. Therefore, the pathwidth of the underlying graph of the attention graph $\mathcal{G}_t$ is at most $|B| + 2 = 3\ell + 2$, which is clearly a function of ℓ.

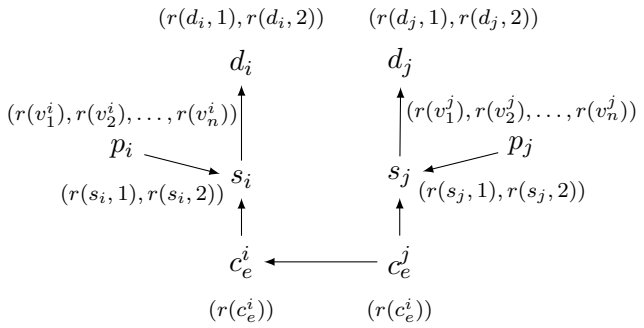


Figure 4: The vertex selection gadget for colors i and j and the certification gadget for a forbidden edge e with endpoints of colors i and j in the constructed instance.

u	$r(v)$	$r(d_h,t)$	$r(s_h,t)$	$r(c_{e'}^h)$
d_i	0	0	0	0
p_i	0	0	0	0
s_i	2	1	0	0
c_e^i	$\begin{cases} 2 & v \in e \cap V_i \\ 1 & \text{o.w.} \end{cases}$	0	1	$\begin{cases} 3 & e = e' \wedge i \neq h \\ 0 & \text{o.w.} \end{cases}$

Table 3: Table of utility functions, where $t \in \{1,2\}$, e and e' are forbidden edges, and $i, h \in [\ell]$.

Correctness We show that the instance of Multicolored Clique is a "yes"-instance if and only if the constructed instance of ERSA is a "yes"-instance. Note that under the initial allocation, every selector s_i envies the corresponding dummy d_i since $u_{s_i}(\pi(s_i)) = 0 < u_{s_i}(\pi(d_i)) = 2$. For each forbidden edge e, both certification agents c_e^i and c_e^j are envious because $u_{c_e^i}(\pi(c_e^i)) = u_{c_e^j}(\pi(c_e^j)) = 0 < u_{c_e^i}(\pi(s_i)) = u_{c_e^j}(\pi(s_j)) = 2$.

$\Rightarrow$: Suppose there is a clique C of size ℓ in G such that C contains one vertex from each color. Based on C, we construct a sharing δ such that after this sharing all selectors become non-envious and for each forbidden edge e exactly one agent from $\{c_e^i, c_e^j\}$ becomes envious. The remaining agents stay non-envious after applying δ.

For each color i, without loss of generality, let $v_1^i \in C$. Then we let provider p_i share resource $r(v_1^i)$ to selector s_i in δ . Hence, each selector s_i receives a resource of utility 2 and they become non-envious. If the input graph G is complete (as a multi-partite graph), then δ is a solution to the constructed instance and the argument for this case is over. Indeed, in such a case there is no forbidden edge for G so that there is no copy of the certificate gadget in the ERSA instance. This in turn implies that applying δ results in all agents being non-envious. To complete the argument, we assume that there is at least one forbidden edge $e = \{v_x^i, v_y^j\}$ for G. Since C is a clique, we have that at least one of v_x^i and v_y^j is not

in C. Thus, as per δ constructed so far, the following two events could not happen at the same time:

1. resource $r(v_x^i)$ is shared to selector s_i;
2. resource $r(v_y^j)$ is shared to selector s_j.

Without loss of generality, assume that resource $r(v_x^i)$ is shared to selector s_i and resource $r(v_{y'}^j)$, instead of $r(v_y^j)$, is shared to selector s_j. Then, we let agent c_e^i share resource $r(c_e^i)$ to agent c_e^j in δ. This makes agent c_e^j non-envious, since $u_{c_e^j}(\pi(c_e^j) \cup \{r(c_e^i)\}) = 3 = u_{c_e^j}(\pi(s_j) \cup \{r(v_{y'}^j)\})$. Applying the just-described way of updating δ for each forbidden edge for G, completes constructing δ. Eventually, exactly one certification agent for each forbidden edge is envious and no other agent is envious, thus there are $\binom{\ell}{2}n^2 - |E| = k$ envious agents after applying sharing δ.

$\Leftarrow$: Suppose there is a sharing δ such that only k agents are envious after this sharing. First, for each forbidden edge $e = \{v_x^i, v_y^j\} \notin G$, the only way to make agent c_e^i not envious is to share resource $r(c_e^j)$ from c_e^j to c_e^i. Analogously, the only way to make agent c_e^j not envious is to share resource $r(c_e^i)$ from c_e^i to c_e^j. Since there could be at most one sharing between c_e^i and c_e^j, at least one agent of c_e^i and c_e^j will always be envious. Since $k = \binom{s}{2}n^2 - |E|$ is exactly the number of all forbidden edges, it must be that after sharing δ no selector is envious and for each forbidden edge e, exactly one agent from $\{c_e^i, c_e^j\}$ is envious. Based on this fact, we show in the following that there is a clique C of size ℓ containing one vertex from each color.

To make selector s_i not envious, s_i has to receive a resource from the provider p_i. Without loss of generality, for each $i \in [\ell]$, let $r(v_1^i)$ be the resource shared to s_i. We claim that $C = \{v_1^1, v_1^2, \ldots, v_1^\ell\}$ is the desired clique. Suppose C is not a clique, then there are two vertices v_1^i and v_1^j such that $\{v_1^i, v_1^j\} \notin E$, which means that edge $e = \{v_1^i, v_1^j\}$ is forbidden. Since every selector s_i shares with provider p_i, agents c_e^i and c_e^j can only share with each other, which leads to increasing their utility to at most 3. However, after such a sharing $u_{c_e^i}(\Pi^\delta(s_i)) = u_{c_e^i}(\pi(s_i) \cup \{r_{x_i}^i\}) = 4$ and $u_{c_e^j}(\Pi^\delta(s_j)) = u_{c_e^j}(\pi(s_j) \cup \{r_{x_j}^j\}) = 4$ so that both c_e^i and c_e^j are envious. This, however, contradicts the fact that at most one of c_e^i and c_e^j is envious in δ. Therefore, C is a clique of size ℓ containing one vertex from each color. □

5.3 Reducing Envy for Identical Agents

We proceed by studying the natural special case where all agents are "identical." That means agents have unanimous utility functions (all agents have the same utility function) and may even have the same social neighborhoods (so everyone may envy everyone else and everyone may share with everyone else). Already in this constrained setting of homogeneous agents allocation problems frequently become hard (Bouveret & Lang, 2008). This is why this scenario also attracts quite some attention in the fair allocation literature (Nguyen et al., 2013; Biswas & Barman, 2018; Barman et al., 2018; Bouveret & Lang, 2008). Here, we focus on cliques that naturally model small, dense communities and allow for convenient comparison with the classical setting of indivisible, non-shareable resources (where the attention graph is implicitly assumed to be a bidirectional clique).

Theorem 3 already shows that restricting the attention graph and the sharing graph to be cliques is not enough to make ERSA solvable in polynomial time. However, assuming unanimous utility functions, Theorem 7 shows that restricting the attention graph to be a bidirectional clique alone yields polynomial-time solvability. The idea behind the proof of Theorem 7 is that non-envious agents are exactly those with the highest utility, so we can guess the highest utility and then reduce the problem to a bounded number of Maximum Matching.

Theorem 7. *For n agents, m resources, unanimous utility functions, and the attention graph being a bidirectional clique,* ERSA *is solvable in $O(nm\rho)$ time, where ρ is the time needed to find a maximum matching in a graph with at most n vertices and at most n^2 edges.*

Proof. Let u be the utility function for all agents and let N be the set of all non-envious agents after a sharing δ. For every two agents $a_i, a_j \in N$, since the attention graph is a clique, we have that $u(\Pi^\delta(a_i)) \geq u(\Pi^\delta(a_j))$ and $u(\Pi^\delta(a_j)) \geq u(\Pi^\delta(a_i))$; hence $u(\Pi^\delta(a_i)) = u(\Pi^\delta(a_j))$. Denote the utility of all agents in N by u^*. Clearly, $u(\Pi^\delta(a_x)) < u^*$ for every agent $a_x \notin N$. Thus, after applying any sharing allocation, the set of non-envious agents is exactly the set of agents who have the highest utility. Based on this observation, it suffices to sweep through all possible values u^* of the target utility and compute the largest number of agents with utility u^* after some sharing.

We first show that the number of different values u^* of the target utility is $O(nm)$. Each agent can get at most one resource through a simple 2-sharing and there are overall m different resources, so each agent could end up with at most m different utility values for its bundle. Since we have n agents, we get the desired upper bound $O(nm)$. Let S be the set of all these utilities, that is, $S = \{u^* \mid u^* = u(\pi(a_i)) + u(r),\, a_i \in \mathcal{A},\, r \in \mathcal{R} \setminus \pi(a_i)\}$. Let $u_0 = \max_{i \in \mathcal{A}} u(\pi(a_i))$ be the largest utility of some agent before a sharing. We only need to consider utility values in $S' = \{u^* \in S \mid u^* \geq u_0\}$ as each target utility u^* should be at least u_0.

We now show that computing the largest number of agents that can have utility u^* after a sharing δ can be reduced to Maximum Matching. Let us fix some value of $u^* \geq u_0$. We first find the set N_0 of agents who already have utility u^* before sharing, that is, $N_0 = \{a_i \in \mathcal{A} \mid u(\pi(a_i)) = u^*\}$. Observe that $N_0 = \emptyset$ if $u^* > u_0$. Then we construct a graph $G = (\mathcal{A}, E)$ over the agents such that an edge $e = \{a_i, a_j\}$ belongs to E if one of agents a_i and a_j can increase its utility to exactly u^* as a result of a sharing between a_i and a_j. Formally, $e = \{a_i, a_j\} \in E$ if $\{a_i, a_j\} \in \mathcal{E}_\mathrm{s}$, $a_i \notin N_0$, and there exists a resource $r \in \pi(a_j)$ such that $u(\pi(a_i)) + u(r) = u^*$. By computing a maximum matching in G, we can find the largest number of agents in $\mathcal{A} \setminus N_0$ who can increase their utilities to u^* through a sharing. Eventually, we have $O(nm)$ runs of Maximum Matching, which yields the claimed running time $O(nm\rho)$. □

We complement Theorem 7 by showing in Theorem 8 that identical utility functions having only two values, identical utility values for all resources, and the sharing graph being a clique are not sufficient to make ERSA solvable in polynomial time when the attention graph can be arbitrary. So, Theorem 7 and Theorem 8 show an interesting and very stark contrast between the influence of the completeness of the attention graph and the sharing graph on the problem's computational complexity.

Theorem 8. ERSA *is* NP*-hard even if the sharing graph is a clique, the agents have a unanimous utility function taking only two values, all resources have the same utility for all agents, and the maximum initial bundle size is one. For the same constraints,* ERSA *is* W[1]*-hard with respect to the parameter "number of resources."*

Proof. We give a polynomial-time many-one reduction from Clique, where we are given an undirected graph together with an integer ℓ and the question is whether there is a set of ℓ mutually connected vertices. Clique is known to be NP-hard, and W[1]-hard with respect to the number of vertices in a sought clique (Downey & Fellows, 2013). To this end, we fix an instance $I = (G, \ell)$ of Clique, where $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{e_1, e_2, \ldots, e_m\}$. For brevity, let $\tilde{\ell} \coloneqq \binom{\ell}{2}$ be the number of edges in a clique of size ℓ. Without loss of generality, we assume that $4 \leq \ell < n$ and $\tilde{\ell} \leq m$. We build an instance I^* of ERSA corresponding to I as follows.

For each vertex $v_i \in V$ we create a *vertex agent* $a(v_i)$ and for each edge $e_i \in E$ we create an *edge agent* $a(e_i)$. We also add $2m$ *dummy agents* $\{d_1, d_2, \ldots, d_{2m}\}$ and $\tilde{\ell}$ *happy agents* $\{h_1, h_2, \ldots, h_{\tilde{\ell}}\}$. We introduce $\tilde{\ell}$ resources $r_1, r_2, \ldots, r_{\tilde{\ell}}$. In the initial allocation π, the resources are given to the happy agents such that each happy agent gets one resource. The resources are indistinguishable for the agents, that is, each agent gives all of them the same utility value; for convenience, we fix this utility to be one. The sharing graph is a clique. In the attention graph $\mathcal{G}_{\mathrm{t}}$, all edge agents have outgoing arcs to all happy agents, each vertex agent $a(v)$, $v \in V$, has an outgoing arc to every edge agent corresponding to an edge incident to v, and all dummy agents have outgoing arcs to all vertex agents, The described ingredients, together with the desired maximum number of envious agents $k \coloneqq m - \tilde{\ell} + \ell$, build an instance I^* of ERSA corresponding to Clique instance I. The whole construction can be done in polynomial time, and we use exactly $\tilde{\ell} = \binom{\ell}{2}$ resources.

Observe that all edge agents are envious under π. Since vertex agents and dummy agents do not pay attention to happy agents, no more agents are envious. Thus, there are exactly $m > k$ envious agents under π. We now show that instance I of Clique is a "yes"-instance if and only if instance I^* of ERSA is a "yes"-instance.

$\Rightarrow$: Assume that there exists a clique of size ℓ in G. Let us call each agent corresponding to an edge in the clique a *clique agent*; by definition, there are exactly $\tilde{\ell}$ of them. We construct a desired sharing for I^* by giving every clique agent a resource. Indeed, it is achievable since we have exactly $\tilde{\ell}$ resources, which can be shared by the happy agents with exactly $\tilde{\ell}$ distinct clique agents. After such a sharing, all clique agents do not envy any more. Observe that the sharing made ℓ agents corresponding to the vertices of the clique envious. Thus, the total number of envious agents is decreased by $\tilde{\ell} - \ell$. This gives exactly the desired number k, resulting in I^* being a "yes"-instance.

$\Leftarrow$: Let δ be a sharing such that there are at most k envious agents under the sharing allocation Π^δ. Since the sharing graph is a clique, happy agents could share resources with any one of the remaining agents. However, if a happy agent shares with a vertex agent, then all $2m$ dummy agents will envy this vertex agent. Since there are only $\tilde{\ell}$ resources, at least $2m - \tilde{\ell} \geq m > k$ dummy agents will be envious at the end. Hence, we can assume that no vertex agent is involved in δ. Consequently, no dummy agent will envy as they only look at vertex agents. Hence, without loss of generality, we can additionally assume that

no dummy agent is involved in δ. Then all sharings in δ are restricted to edge agents and happy agents.

Now, we show that δ actually encodes a clique of size ℓ in G. When an edge agent gets a resource through sharing, it stops being envious, however, all vertex agents that looks at this edge agent become envious. So, an ultimate goal for an optimal sharing is to share resources to the highest possible number of edge agents that are connected with an arc with as few vertex agents as possible. Formally, let S be the set of edge agents who get shared resources in δ and T the set of the corresponding vertex agents who become envious as a result of the sharing. Denote $s = |S|$ and $t = |T|$. Naturally, $1 \leq s \leq \tilde{\ell}$ and $t \geq 2$. Moreover, by definition we have $s \leq \binom{t}{2}$. Then, the number of envious agents after the sharing in the solution is at least $m - s + t$ and this number should be at most $k = m - \tilde{\ell} + \ell$, that is,

$$m - s + t \leq m - \tilde{\ell} + \ell \Rightarrow s - t \geq \tilde{\ell} - \ell.$$

Since $\tilde{\ell} = \binom{\ell}{2}$ and $s \leq \binom{t}{2}$, if $\ell > t \geq 2$, then

$$s - t \leq \binom{t}{2} - t < \binom{\ell}{2} - \ell = \tilde{\ell} - \ell,$$

which contradicts that $s - t \geq \tilde{\ell} - \ell$. Hence $t \geq \ell$. Since $s \leq \tilde{\ell}$, the only way to satisfy $s - t \geq \tilde{\ell} - \ell$ is that $s = \tilde{\ell}$ and $t = \ell$. Thus, the corresponding vertices of T form a clique of size ℓ in G. □

6. Extensions

We introduce two natural extensions of our model, both of which describe costs which can be incurred by sharing—either for the central authority or agents. For each extension, we discuss which results can easily be adapted to cover it. Finally, we formally present how to modify the proofs of the relevant results.

6.1 Extension 1: Loss by Sharing

So far, we have assumed that agents get the full utility of the shared resources. This does not hold for situations in which sharing causes more inconvenience than owning a resource alone. Nonetheless, most of our algorithms can be easily adapted to deal with (computational) issues that arise in such situations. Consider the case when agents get only a fraction of the full utility from shared resources. Then our algorithms for improving utilitarian welfare (Theorem 1) and egalitarian welfare (Proposition 1) still work with minor changes. Regarding reducing envy, it might be that after sharing agents lose some utility and thus become envious. Again, our algorithms for tree-like graphs (Theorem 6) or identical agents (Theorem 7) still work with minor changes. The only exception is the FPT algorithm for few agents (Theorem 4): Whether this FPT algorithm can be adapted for Extension 1 is still an open question.

Formally, we introduce two parameters $\alpha, \beta \in [0, 1]$ to quantify the effect that agents do not get the full utility of the shared resources. For every resource r initially assigned to agent a_i in π and shared to a_j under δ, the utility of resource r for a_i is $\alpha \cdot u_i(r)$ and for a_j is $\beta \cdot u_j(r)$. Recall that we refer to $\Pi^{\delta}(a)$ as a *bundle* of a. Now we refer to $\Pi^{\delta}_{+}(a)$ as the set

of resources shared to a by other agents, $\Pi^{\delta}_{-}(a)$ the set of resources shared to other agents by a, and $\Pi^{\delta}_{0}(a)$ the set of remaining (not shared) resources. Then the utility of agent a under Π^{δ} is

$$u_i(\Pi^{\delta}(a_i)) = \sum_{r \in \Pi^{\delta}_{0}(a)} u_i(r) + \sum_{r \in \Pi^{\delta}_{-}(a)} \alpha u_i(r) + \sum_{r \in \Pi^{\delta}_{+}(a)} \beta u_i(r).$$

Later, in Section 6.4, we show how to incorporate this extension to all of our algorithmic results (Theorem 1, Proposition 1, Theorem 6, Theorem 4, and Theorem 7). Notice that the original setting corresponds to the case with $\alpha = \beta = 1$.

6.2 Extension 2: Cost of Sharing

It is also natural to assume that the central authority needs to pay some cost for each sharing to incentivize agents to share resources. In this case, there is a limited budget that the central authority can spend and the goal is to improve the allocation through sharings whose total cost does not exceed the budget. So far, our model was not capable of modeling the described scenario. However, except for Theorem 1, all our algorithms still work for this generalized setting with minor changes.

More precisely, for a simple 2-sharing δ, we introduce a *sharing cost* $c_{\mathrm{g}} \colon \mathcal{E}_{\mathrm{s}} \to \mathbb{N}$ and a budget $B \in \mathbb{N}$ for the central authority to incentivize sharing. Notice that the sharing cost is defined for each pair of agents. The cost of a sharing δ for the central authority is

$$c(\delta) = \sum_{\delta(\{a_i, a_j\}) \neq \emptyset} c_{\mathrm{g}}(\{a_i, a_j\}).$$

Now the goal of the central authority is to find a sharing δ with $c(\delta) \leq B$ to improve the allocation.

Later, in Section 6.4, we show how to take care of this extension in the proofs of Proposition 1, Theorem 4, Theorem 6, and Theorem 7. Notice that the original setting corresponds to the case with $B = \sum_{\{a_i, a_j\} \in \mathcal{E}_{\mathrm{s}}} c_{\mathrm{g}}(\{a_i, a_j\})$, that is, $c(\delta) \leq B$ holds for any sharing δ.

6.3 Useful Matching-Related Computational Problems

We define two variants of finding a maximum weight matching and show that they are solvable in polynomial time; to the best our knowledge, these variants are not present in the literature. We use one of them to show polynomial-time solvability of the other one, which we then use as a subroutine in one of the subsequent algorithms.

Given an undirected weighted graph G with a weight function $w \colon E \to \mathbb{N}$ and two integers $k_1, k_2 \in \mathbb{N}$, Size-Bounded Maximum Weight Matching (SBMWM) asks whether there is a matching M in G such that $|M| \leq k_1$ and $w(M) \geq k_2$. For the same input, Weight-Bounded Maximum Matching (WBMM) asks whether there is a matching M in G such that $w(M) \leq k_1$ and $|M| \geq k_2$.

Lemma 3. SBMWM *and* WBMM *are solvable in* $O(|V|^3)$ *time.*

Proof. The SBMWM problem is a specialization of the Constrained Matching Problem (CMP) defined by Plesník (1999) [Theorem 1] who also showed it to be solvable

in $O(|V|^3)$ time. In CMP, given a weighted graph G, two integers ℓ (the lower bound) and u (the upper bound), and a set of required vertices R, the task is to find a maximum weight matching of cardinality at least ℓ and at most u such that each vertex $v \in R$ is adjacent to at least one edge in the matching. It is easy to see, that to solve an instance (G, w, k_1, k_2) of SBMWM, one can solve an instance of CMP with the same graph, lower bound $\ell := 0$, upper bound $u := k_1$, and $R := \emptyset$ and compare the weight of the output matching of the latter with k_2. Hence, SBMWM is solvable in $O(|V|^3)$ time.

To show the claim for WBMM[4] let us consider its instance (G, w, k_1, k_2) with $G = (V, E)$. Let $W := \max_{e \in E} w(e)$ and let $w' \colon E \to \mathbb{N}$ be a new weight function such that $w'(e) := W - w(e)$ for each $e \in E$. Note that $w(e) \geq 0$ and $w'(e) \geq 0$ for each $e \in E$. Since $w(e) \geq 0$, G admits a matching M^* with $w(M^*) \leq k_1$ and $|M^*| \geq k_2$ if and only if there is a matching $M \subseteq M^*$ in G such that $w(M) \leq k_1$ and $|M| = k_2$. When $|M| = k_2$, then $w(M) = Wk_2 - w'(M)$, and hence $w(M) \leq k_1$ if and only if $w'(M) \geq Wk_2 - k_1$. Thus, eventually, the problem is to decide whether, using the new weight function w', graph G has such a matching M with $|M| = k_2$ and $w'(M) \geq Wk_2 - k_1$. Since $w'(e) \geq 0$, this is equivalent to decide whether there is a matching M with $|M| \leq k_2$ and $w'(M) \geq Wk_2 - k_1$, which exactly yields an instance of SBMWM which is solvable in $O(|V|^3)$ time, as we have shown in the previous paragraph. □

6.4 Respective Proofs' Modifications

In the ensuing paragraphs, we list the necessary modifications of our results to make them work correctly for the aforementioned extensions.

Proof of Theorem 1 for Extension 1 We adapt the proof of Theorem 1 given earlier to work for Extension 1. To this end, we only need to change the weights of edges in the constructed graph as follows. The weight of the edge between $v_{i_1}^{j_1}$ and $v_{i_2}^{j_2}$ is now defined as

$$\max\{\beta u_{i_1}(r_{j_2}) - (1-\alpha)u_{i_2}(r_{j_2}), \beta u_{i_2}(r_{j_1}) - (1-\alpha)u_{i_1}(r_{j_1})\}$$

instead of $\max\{u_{i_1}(r_{j_2}), u_{i_2}(r_{j_1})\}$, i.e., the weight is the increased utilitarian social welfare in the new setting. Accordingly, in the "⇐:" part, for each non-dummy edge $(v_{i_1}^{j_1}, v_{i_2}^{j_2}) \in M$, we set $\delta(a_{i_1}, a_{i_2}) = r_{j_1}$ if the increased utilitarian social welfare by sharing r_{j_1} to agent a_{i_2} is no smaller than that of sharing r_{j_2} to agent a_{i_1}, and $\delta(a_{i_1}, a_{i_2}) = r_{j_2}$ otherwise. Then, using the same arguments, we can show that there is a b-bounded 2-sharing δ such that $\mathrm{usw}(\Pi^\delta) \geq k$ if and only if there is matching M in graph G with weight $\sum_{e \in M} w(e) \geq k - \mathrm{usw}(\pi) + P$, and thus, the problem can be reduced to Maximum Weighted Matching. □

1-UWSA with Extension 2 is weakly NP-hard Theorem 1 is the only algorithmic result which cannot be extended for Extension 2. We reduce from the weakly NP-complete Partition problem (Garey & Johnson, 1979, SP12). Given a multiset of positive integers $S = \{u_1, \ldots, u_t\}$, Partition asks whether there is a subset $S' \subseteq S$ such that $\sum_{u_i \in S'} u_i = \sum_{u_i \in S \setminus S'} u_i$.

4. To the best of our knowledge, there is no explicit algorithm for solving WBMM in the matching literature yet. We, however, note that Plesník (1999) [Theorem 6] showed polynomial-time solvability for a very similar problem of finding minimum cost edge-covers of a graph. Yet, this variant does not require that the edges of the edge cover should form a matching (that is, that they should be non-adjacent).

Given an instance $S = \{u_1, \ldots, u_t\}$ of Partition, for each integer $u_i \in S$, we create an input instance with one pair of agents a_i, a_i', one resource r_i owned by a_i, and one edge $\{a_i, a_i'\}$ in the sharing graph. We assign utility u_i to resource r_i for agent a_i' and sharing costs u_i for agent a_i sharing r_i with a_i'.

Now, it is easy to verify that asking whether there is a way to increase the welfare by at least $B = 1/2 \cdot \sum_{u_i \in S} u_i$ at a total cost of at most B, is equivalent to ask whether there is a subset $S' \subseteq S$ such that $\sum_{u_i \in S'} u_i = \sum_{u_i \in S \setminus S'} u_i$. □

Proof of Proposition 1 for Extensions 1 and 2 We adapt the original proof of Proposition 1 to work for the case with both Extension 1 and Extension 2. Similarly as before, we partition the set $\mathcal{A}$ of agents into two sets $\mathcal{A}_k^+$ and $\mathcal{A}_k^-$ containing, respectively, the agents with their bundle value under π at least k and smaller than k. When constructing the graph $G_k = (\mathcal{A}_k^+, \mathcal{A}_k^-, E_k)$, for two agents $a_i \in \mathcal{A}_k^+$ and $a_j \in \mathcal{A}_k^-$ that are neighbors in the sharing graph $\mathcal{G}_\text{s}$, we add an edge $e = \{a_i, a_j\}$ to E_k if a_i can share a resource with a_j to raise the utility of the latter to at least k given that after this sharing the utility of a_i is at least k. This holds true, if there is a resource $r \in \pi(a_i)$ such that $u_j(\pi(a_j)) + \beta u_j(r) \geq k$ and $u_i(\pi(a_i)) - (1-\alpha)u_i(r) \geq k$. In addition, for each edge $e = \{a_i, a_j\} \in E_k$ we assign it weight $c_\text{g}(\{a_i, a_j\})$. With similar arguments, we have that there is a simple 2-sharing δ with $c(\delta) \leq B$ and $\text{esw}(\Pi^\delta) \geq k$ if and only if there is matching M in graph G_k with $\sum_{e \in M} c_\text{g}(e) \leq B$ and $|M| \geq |\mathcal{A}_k^-|$. Thus, we just need to check whether there is matching M in graph G_k with $\sum_{e \in M} c_\text{g}(e) \leq B$ and $|M| \geq |\mathcal{A}_k^-|$, which is an instance of WBMM and is solvable in polynomial time according to Lemma 3. □

Proof of Theorem 4 for Extension 2 When there is a cost of sharing, notice that the cost of every realization δ of a configuration M in Algorithm 1 is fixed: $c(\delta) = c(M)$. Therefore, it suffices to check first whether the cost of the guessed configuration M is at most B. □

Open: Is ERSA **with Extension 1** FPT **with respect to** n**?** We briefly explain why the proof of Theorem 4 does not work for Extension 1. In the original proof, when checking the existence of a desired matching for each guessed pair (C, M), we first keep deleting resources whose values are either too low for the receiver or too high for agents watching the receiver (Step 1), and afterward we can simply select for each receiver the most valuable remaining resource (Step 2). This idea works because after deleting unsuitable resources, Step 2 ensures that no agent in the target set C would become envious. However, in Extension 1, donors lose utility after sharing, and it's possible for these donors to be part of the target set C. This complicates the problem as the final utilities of these donors depend on the final sharing we choose, whereas in the original setting, their utilities are fixed. As a result, it is challenging to determine which resources to delete in Step 1. Specifically, it is difficult to identify resources whose values are too high for donors whose utilities are not fixed.

Proof of Theorem 6 for Extensions 1 and 2 We adapt the original proof of Theorem 6 to work for the case with both Extension 1 and Extension 2. The adaption for Extension 1 is trivial: The only change is that when computing the set of envious agents under a sharing, we need to compute the utility of agents according to the definition in Extension 1.

For Extension 2, considering an upper bound for the sharing cost, we define a new function F extending the original function f. Recall that the original function f is defined as:

$$f[t, b, S] \coloneqq \min\{\mathrm{Env}_{[V_t]}(\delta) \mid (\delta \text{ realizes } b) \wedge (\mathrm{Env}_{[V_t]}(\delta) \cap X_t = S)\}.$$

Extending f, we define F by adding a new input variable to control the number of envious agents and taking the sharing cost as the minimization objective. So, for every node X_t, every bundle configuration $b \in B_t$, every subset $S \subseteq X_t$, and every $e \in [0, k]$, the definition of F reads as follows:

$$F[t, b, S, e] \coloneqq \min\{c(\delta) \mid (\delta \text{ realizes } b) \wedge (\mathrm{Env}_{[V_t]}(\delta) \cap X_t = S) \wedge (|\mathrm{Env}_{[V_t]}(\delta)| = e)\}.$$

In words, $F[t, b, S, e]$ is the minimum cost of a sharing δ such that δ realizes the bundle configuration b on X_t and such that in the subinstance induced by the agents in V_t under sharing δ the set of envious agents from X_t is S and the number of envious agents in V_t is e. Note that if under b there is some agent $v \in X_t$ who receives a resource from some other agent $v' \notin V_t$, then the cost for this sharing pair will be included in each sharing realizing b. Taking $r \in V(T)$ as the root node of T, now our goal is to compute $\min\{F[r, \pi, \emptyset, e] \mid e \in [0, k]\}$. Next we discuss the changes for computing the four possible types of nice tree decomposition nodes: leaf node, introduce node, forget node, and join node.

Leaf node. Suppose t is a leaf node of T, then $V_t = X_t = \emptyset$ and $B_t = \emptyset$. We get:

$$f[t, \pi, \emptyset, e] = \begin{cases} 0 & \text{if } e = 0, \\ +\infty & \text{if } e > 0. \end{cases}$$

Introduce node. Suppose t is an introduce node with a child t' such that $X_t = X_{t'} \cup \{v\}$ for some $v \notin X_{t'}$. Recall that our old function f for an introduce node is computed as:

$$f[t, b, S] = \min\{|S \setminus S^*| + f[t', b[X_{t'}], S^*] \mid S \setminus E \subseteq S^* \subseteq S \cap X_{t'}\}.$$

Now, we have

$$\begin{aligned} F[t, b, S, e] = \min\{&c(b[v]) \\ &+ F[t', b[X_{t'}], S^*, e^*] \mid (S \setminus E \subseteq S^* \subseteq S \cap X_{t'}) \wedge (e = e^* + |S \setminus S^*|)\}, \end{aligned}$$

where $c(b[v])$ is the sharing cost required to ensure that agent v has bundle $b(v)$. That is, depending on the initial allocation π, $c(b[v]) = 0$ if $b(v) = \pi(v)$ and otherwise $c(b[v]) = c_{\mathrm{g}}(\{v, v'\})$ where v' is the agent who shares one resource with v to ensure $b(v)$. So compared with the old computation, we introduce two major changes. First, we alter the increase of the objective by substituting the increase of the number $|S \setminus S^*|$ of envious agents by the increase $c(b[v])$ of the cost. Then, we add the requirement related to the number $e = e^* + |S \setminus S^*|$ of envious agents to the condition restricting considered (previous) values of F.

Forget node. Suppose t is a forget node with a child t' such that $X_t = X_{t'} \setminus \{w\}$ for some $w \in X_{t'}$. Recall that our old function f for a forget node is computed as:

$$f[t, b, S] = \min\{f[t', b^*, S^*] \mid (b^*[X_t] = b) \wedge (S \subseteq S^* \subseteq S \cup \{w\})\}.$$

Now, for every $b \in B_t$, $S \subseteq X_t$ and $e \in [0, k]$, we have that:

$$F[t, b, S, e] = \min\{F[t', b^*, S^*, e] \mid (b^*[X_t] = b) \wedge (S \subseteq S^* \subseteq S \cup \{w\})\}.$$

Join node. Suppose t is a join node with children t_1 and t_2 such that $X_t = X_{t_1} = X_{t_2}$. Recall that our old function f for a join node is computed as:

$$f[t, b, S] = \min\{f[t_1, b, S_1] + f[t_2, b, S_2] - |S_1 \cap S_2| \mid S_1 \cup S_2 = S\}.$$

Now, for each $b \in B_t$, $S \subseteq X_t$ and $e \in [0, k]$, we have that:

$$\begin{aligned} F[t, b, S, e] = \min\{&F[t_1, b, S_1, e_1] \\ &+ F[t_2, b, S_2, e_2] - c(b[X_t]) \mid (S_1 \cup S_2 = S) \wedge (e = e_1 + e_2 - |S_1 \cap S_2|)\}. \end{aligned}$$

where $c(b[X_t])$ is the sharing cost to ensure the bundle configuration b for agents in X_t. We subtract $c(b[X_t])$ because it is included in both $F[t_1, b, S_1, e_1]$ and $F[t_2, b, S_2, e_2]$. □

Proof of Theorem 7 for Extension 1 We adapt the earlier version of the proof to work for the case with Extension 1, where agents only get a fraction of utility for shared resources.

If $\alpha = 1$, then no agent will lose utility after sharing, and we just need to change the edge set E as follows. Recall that N_0 is defined as the set of agents who already have utility u^* before sharing. For each pair of distinct agents a_i and a_j, we add $e = \{a_i, a_j\}$ to E if $\{a_i, a_j\} \in \mathcal{E}_\text{s}$ and (1) $a_i \notin N_0$ and there is a resource $r \in \pi(a_j)$ such that $u(\pi(a_i)) + \beta u(r) = u^*$. Naturally, in case we add two edges between the same pair of agents, we only keep one copy of the edge.

If $\alpha < 1$, the problem becomes more complex as in this case the target utility u^* could be smaller than $u_0 = \max_{i \in \mathcal{A}} u(\pi(a_i))$, the largest utility of some agent before a sharing. Nevertheless, one can easily verify that the number of different values for u^* is still bounded by $O(nm)$. Next we show how to solve the case with $u^* < u_0$. If $\beta = 0$, then no agent can increase its utility by sharing and we need to check whether we can find a sharing such that all agents who originally have utility higher than u^* can decrease their utility to be at most u^*, and meanwhile maximizing the number of them with utility exactly u^* after sharing. To this end, we build a bipartite graph $G = (V_1 \cup V_2, E)$ with one side V_1 representing agents who have utility higher than u^* and the other side V_2 representing the remaining agents. For each pair of agents $\{a_i, a_j\}$ with $a_i \in V_1$ and $a_j \in V_2$, if $\{a_i, a_j\} \in \mathcal{E}_\text{s}$ and there exists a resource $r \in \pi(a_i)$ such that $u(\pi(a_i)) - (1 - \alpha)u(r) \leq u^*$, we add $e = \{a_i, a_j\}$ to E. In addition, we assign the edge e a weight $Q + 1$ if $u(\pi(a_i)) - (1 - \alpha)u(r) = u^*$ and Q if $u(\pi(a_i)) - (1 - \alpha)u(r) < u^*$, where $Q > |\mathcal{A}|^2$ is a large number. Then we compute a maximum weighted matching in G; suppose it has weight W. If $W < Q|V_1|$, then it is impossible to make the highest utility to be u^* after any sharing. Otherwise, the maximum weighted matching indicates a sharing such that after the sharing all agents will have utility at most u^*. Moreover, the number of agents with utility exactly u^* is $|N_0| + W - Q|V_1|$, where N_0 is the set of agents who had utility u^* before sharing.

Finally, we solve the case with $\alpha < 1$ and $\beta > 0$. We partition all agents into three groups N_+, N_0, N_- according to their utilities before sharing, where $N_+ = \{a_i \in \mathcal{A} \mid u(\pi(a_i)) > u^*\}$, $N_0 = \{a_i \in \mathcal{A} \mid u(\pi(a_i)) = u^*\}$, and $N_- = \{a_i \in \mathcal{A} \mid u(\pi(a_i)) < u^*\}$. Notice

that there is no benefit to make an agent in N_0 participate in a sharing. Such a sharing can make its utility either higher than u^* or smaller than u^*. The former is unwanted as we aim at u^* being the maximum utility. Regarding the latter, as a result of sharing at most one other agent may get utility exactly u^*, so it does not help to increase the total number of non-envious agents. Hence, we build an edge-weighted graph $G = (N_+ \cup N_-, E)$. For each pair $\{a_i, a_j\}$ of agents with $a_i \in N_+$ and $a_j \in N_-$, if $\{a_i, a_j\} \in \mathcal{E}_s$ and they can share a resource such that both of them have utility at most u^* after sharing, then we add an edge $e = \{a_i, a_j\}$ with weight $Q+i$ to E, where $Q > |\mathcal{A}|^2$ is a large number and $i \in \{0, 1, 2\}$ is the number of agents from $\{a_i, a_j\}$ who have utility exactly u^* after the sharing. In addition, for each pair of agents both from N^-, if they can share a resource such that one of them has utility exactly u^* after a sharing, then there is an edge with weight 1 between them. Then we compute a maximum weighted matching in G; suppose it has weight W. If $W < Q|N_+|$, then it is impossible to make the highest utility to be u^* after any sharing. Otherwise, the maximum weighted matching indicates a sharing such that after the sharing all agents have utility at most u^* and the number of agents with utility exactly u^* is $|N_0| + W - Q|N_+|$. □

Proof of Theorem 7 for Extension 2 For Extension 2 where there is a cost for sharing, we convert the graph $G = (\mathcal{A}, E)$ in the original proof of Theorem 7 into a weighted graph by assigning each edge $e \in E$ weight $c_g(e)$. Then, we compute a maximum-size matching with weight at most B in G, that is, we solve Weight-Bounded Maximum Matching, which can be done in polynomial time according to Lemma 3. The matching size is then the largest number of agents in $\mathcal{A} \setminus N_0$ who can increase their utilities to u^* through a sharing. □

Open: Is ERSA **with Extensions 1 and 2 polynomial-time solvable for unanimous utility functions and the attention graph being a bidirectional clique?** Notice that to solve the problem with Extension 1, we need to find a maximum weighted matching in an edge-weighed graph, while Extension 2 brings another edge weight function for which we need to minimize when searching for the matching. Thus the problem with both Extension 1 and 2 has two different weight-like constraints similar to the Knapsack Problem, which makes it difficult to design polynomial-time algorithms. On the other hand, the weight related to Extension 1 uses only 4 values, which makes it difficult to show NP-hardness.

7. Conclusion and Discussion

We brought together two important topics—fair allocation of resources and resource sharing. We have introduced a basic model, in which we assumed agents are initially endowed with resources, which can then be shared by neighbors in a social network in a way that agent participates in a bounded number of sharing.

Providing a new model, we contributed to a recent line of research aiming at achieving fairness taking an approach that does not relax the concept of envy but utilizes social aspects to improve collective wellness. Importantly, our model is suitable for scenarios for which a standard approach—of computing a fair allocation from scratch—cannot be used because modifying the initial endowment of agents is undesired or impossible (like our knowledge sharing example).

Our model, albeit basic, has proven to lead to challenging computational problems. We shed light at their fundamental computational complexity limitations (in the form of computational hardness) and provided generalizable algorithmic techniques (as mentioned in Section 6). The latter ones mark scenarios (like improving the utilitarian and egalitarian social welfare; or decreasing the number of envious agents among groups of few agents) for which our results indicate that improving resource allocations by sharing in pairs might be applicable in practice.

Finally, there is rich potential for future research exploring our general model of sharing allocations (in its full power described by Definition 3). In what follows, we briefly lay out a few possible follow-up topics. We note that even though below we focus on theoretical research, an empirical study of our algorithms and concepts could also be of interest.

Deepening Parametric Analysis We showed that ERSA is NP-hard even if both input graphs are (bidirectional) cliques but it is polynomial-time solvable if two graphs are the same and have constant treewidth. Based on this, analyzing various combinations of graph classes of the two social networks might be valuable. Some of our reductions (Theorem 5, Theorem 6, Theorem 8) rely on directed attention graphs. For these results, it is interesting to consider the case with undirected attention graphs. For Theorem 5 and Theorem 8, where we study few agents or resources, the case with both graphs being (bidirectional) cliques is also compelling.

Additionally, investigating whether our positive results can be extended to so-called high-multiplicity scenarios (Budish, 2011; Bredereck et al., 2019), where an unbounded number of objects (agents or resources) come from a bounded set of types of objects, is also a challenge from the perspective of computability limits of our model. Such analysis sometimes lead to specialized algorithms that perform quite well in practice.

Beyond 2-Sharing We focused on sharing resources between neighbors in a social network. Yet, there are many scenarios where sharing resources among a large group of agents can be very natural and wanted. If every resource can be shared with everyone, then there is a trivial envy-free allocation. Hence, it is interesting to further study the computational limits of improving allocations under various sharing relaxations (concerning parameters such as number of shared resources, number of agents sharing a resource, etc.).

In the same spirit of extending our model, it is intriguing to study our model dropping the assumption that there is a complete initial allocation given, that is, allocating indivisible but shareable resources to achieve welfare and/or fairness goal. While this model appears simpler than our current one, our model's simplicity lies in that we only need to determine the sharing and the initial allocation constrains the sharing scope.

Strategic Concerns and Robustness We have assumed that all utility values are truthfully reported as well as correct and that the agents need not to be incentivized to share resources. Neither of these assumptions might be justified in some cases—the agents might misreport their utility, the utility values might be slightly incorrect, or a sharing can come at a cost *for agents* (splitting utility from shared resources, as described in Section 6, is an example of the latter). Tackling this kind of issues opens a variety of directions, which includes studying strategic misreporting of utilities, robustness of computed solutions against small utility values perturbations, and finding allocations that incentivize sharing.

Acknowledgments

We dedicate this paper to Rolf, who tragically passed away recently. We are deeply affected by this loss of our co-author, colleague, and advisor. Rolf contributed tremendously to computer science and, in particular, to multivariate algorithms, and should have continued doing so for a long time. The computer science community will build on the foundations he has laid.

A short version (Bredereck, Kaczmarczyk, Luo, Niedermeier, & Sachse, 2022a) appeared in *The Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI '22)*

We are grateful to the anonymous AAAI'22 and Journal of Artificial Intelligence Research (JAIR) reviewers for their insightful comments. This work was started when all authors were with TU Berlin. Andrzej Kaczmarczyk was supported by the DFG project "AFFA" (BR 5207/1 and NI 369/15) and by the European Research Council (ERC). Junjie Luo was supported by the DFG project "AFFA" (BR 5207/1 and NI 369/15), the Singapore Ministry of Education Tier 2 grant (MOE2019-T2-1-045) and the Talent Fund of Beijing Jiaotong University (2023XKRC007).

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 101002854).

References

Abebe, R., Kleinberg, J., & Parkes, D. C. (2017). Fair division via social comparison. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS'17)*, pp. 281–289.

Aziz, H., Bouveret, S., Caragiannis, I., Giagkousi, I., & Lang, J. (2018). Knowledge, fairness, and social constraints. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*, pp. 4638–4645.

Barman, S., Krishnamurthy, S. K., & Vaish, R. (2018). Finding fair and efficient allocations. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC'18)*, pp. 557–574.

Bei, X., Qiao, Y., & Zhang, S. (2017). Networked fairness in cake cutting. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, pp. 3632–3638.

Belk, R., Eckhardt, G., & Bardhi, F. (2019). *Handbook of the Sharing Economy*. Edward Elgar Pub.

Bentham, J. (1823). *An Introduction to the Principles of Morals and Legislation*. Clarendon Press.

Beynier, A., Chevaleyre, Y., Gourvés, L., Harutyunyan, A., Lesca, J., Maudet, N., & Wilczynski, A. (2019). Local envy-freeness in house allocation problems. *Autonomous Agents and Multi-Agent Systems*, *33*, 591–627.

Biswas, A., & Barman, S. (2018). Fair division under cardinality constraints. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'18)*, pp. 91–97.

Bouveret, S., Cechlárová, K., Elkind, E., Igarashi, A., & Peters, D. (2017). Fair division of a graph. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, pp. 135–141.

Bouveret, S., Chevaleyre, Y., & Maudet, N. (2016). Fair allocation of indivisible goods. In Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.), *Handbook of Computational Social Choice*, chap. 13, pp. 311–329. Cambridge University Press.

Bouveret, S., & Lang, J. (2008). Efficiency and envy-freeness in fair division of indivisible goods: Logical representation and complexity. *Journal of Artificial Intelligence Research*, *32*(1), 525–564.

Bredereck, R., Kaczmarczyk, A., Knop, D., & Niedermeier, R. (2019). High-multiplicity fair allocation: Lenstra empowered by n-fold integer programming. In *Proceedings of the 2019 ACM Conference on Economics and Computation (EC'19)*, pp. 505–523.

Bredereck, R., Kaczmarczyk, A., Luo, J., Niedermeier, R., & Sachse, F. (2022a). On improving resource allocations by sharing. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI '22)*, pp. 4875–4883.

Bredereck, R., Kaczmarczyk, A., & Niedermeier, R. (2022b). Envy-free allocations respecting social networks. *Artificial Intelligence*, *305*, 103664.

Brustle, J., Dippel, J., Narayan, V. V., Suzuki, M., & Vetta, A. (2020). One dollar each eliminates envy. In *Proceedings of the 21st ACM Conference on Economics and Computation (EC'20)*, pp. 23–39.

Budish, E. (2011). The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, *119*(6), 1061–1103.

Caragiannis, I., Gravin, N., & Huang, X. (2019). Envy-freeness up to any item with high Nash welfare: The virtue of donating items. In *Proceedings of the 20th ACM Conference on Economics and Computation (EC'19)*, pp. 527–545.

Caragiannis, I., & Ioannidis, S. D. (2022). Computing envy-freeable allocations with limited subsidies. In *Proceedings of the 18th International Conference on Web and Internet Economics (WINE '22)*, pp. 522–539. Springer.

Caragiannis, I., Kurokawa, D., Moulin, H., Procaccia, A. D., Shah, N., & Wang, J. (2019). The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation*, *7*(3), 12:1–12:32.

Chaudhury, B. R., Kavitha, T., Mehlhorn, K., & Sgouritsa, A. (2021). A little charity guarantees almost envy-freeness. *SIAM Journal on Computing*, *50*(4), 1336–1358.

Chevaleyre, Y., Endriss, U., & Maudet, N. (2017). Distributed fair allocation of indivisible goods. *Artificial Intelligence*, *242*, 1–22.

Chevaleyre, Y., Endriss, U., & Maudet, N. (2007). Allocating goods on a graph to eliminate envy. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI'07)*, pp. 700–705.

Cygan, M., Fomin, F. V., Kowalik, Ł., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., & Saurabh, S. (2015). *Parameterized Algorithms*. Springer.

Downey, R. G., & Fellows, M. R. (2013). *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer.

Duan, R., & Pettie, S. (2014). Linear-time approximation for maximum weight matching. *Journal of the ACM*, *61*(1), 1–23.

Duan, R., Pettie, S., & Su, H.-H. (2018). Scaling algorithms for weighted matching in general graphs. *ACM Transactions on Algorithms*, *14*(1), 1–35.

Eiben, E., Ganian, R., Hamm, T., & Ordyniak, S. (2023). Parameterized complexity of envy-free resource allocation in social networks. *Artificial Intelligence*, *315*, 103826.

Friedman, E., Psomas, C.-A., & Vardi, S. (2015). Dynamic fair division with minimal disruptions. In *Proceedings of the 16th ACM Conference on Economics and Computation (EC'15)*, pp. 697–713.

Friedman, E., Psomas, C.-A., & Vardi, S. (2017). Controlled dynamic fair division. In *Proceedings of the 18th ACM Conference on Economics and Computation (EC'17)*, pp. 461–478.

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Gourvès, L., Lesca, J., & Wilczynski, A. (2017). Object allocation via swaps along a social network. In Sierra, C. (Ed.), *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, pp. 213–219.

Halpern, D., & Shah, N. (2019). Fair division with subsidy. In *Proceedings of the 12th International Symposium on Algorithmic Game Theory (SAGT'19)*, pp. 374–389.

He, J., Procaccia, A. D., Psomas, A., & Zeng, D. (2019). Achieving a fairer future by changing the past. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*, pp. 343–349.

Hosseini, H., Sikdar, S., Vaish, R., Wang, H., & Xia, L. (2020). Fair division through information withholding. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20)*, pp. 2014–2021.

Huang, S., & Xiao, M. (2020). Object reachability via swaps under strict and weak preferences. *Autonomous Agents Multi Agent Systems*, *34*(2), 51.

Kloks, T. (1994). *Treewidth, Computations and Approximations*, Vol. 842 of *Lecture Notes in Computer Science*. Springer.

Lange, P., & Rothe, J. (2019). Optimizing social welfare in social networks. In *Proceedings of the 6th International Conference on Algorithmic Decision Theory (ADT'19)*, pp. 81–96.

Lipton, R. J., Markakis, E., Mossel, E., & Saberi, A. (2004). On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC'04)*, pp. 125–131.

Nguyen, T. T., Roos, M., & Rothe, J. (2013). A survey of approximability and inapproximability results for social welfare optimization in multiagent resource allocation. *Annals of Mathematics and Artificial Intelligence*, *68*(1-3), 65–90.

Plesník, J. (1999). Constrained weighted matchings and edge coverings in graphs. *Discrete Applied Mathematics*, *92*(2), 229–241.

Rawls, J. (1971). *A Theory of Justice*. Harvard University Press.

Sandomirskiy, F., & Segal-Halevi, E. (2022). Efficient fair division with minimal sharing. *Operations Research*, *70*(3), 1762–1782.

Schor, J. B., & Cansoy, M. (2019). The sharing economy. In Wherry, F. F., & Woodward, I. (Eds.), *The Oxford Handbook of Consumption*, pp. 51–74. Oxford University Press.

Segal-Halevi, E. (2019). Fair division with bounded sharing. *CoRR*, *abs/1912.00459*.

Segal-Halevi, E. (2022). Redividing the cake. *Autonomous Agents and Multi-Agent Systems*, *36*(1), 14.

Vardi, S., Psomas, A., & Friedman, E. J. (2022). Dynamic fair resource division. *Mathematics of Operations Research*, *47*(2), 945–968.

Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming. *IEEE Software*, *17*(4), 19–25.