Reduction of Economic Inequality in Combinatorial Domains

Endriss, U.

Publication date
2013
Document Version
Final published version
Published in
AAMAS'13

Reduction of Economic Inequality in Combinatorial Domains

Ulle Endriss
Institute for Logic, Language and Computation
University of Amsterdam
ulle.endriss@uva.nl

ABSTRACT

Criteria for measuring economic inequality, such as the Lorenz curve and the Gini index, are widely used in the social sciences but have hardly been explored in Multiagent Systems, even though the significance of other concepts from fair division is widely accepted in the field. In a departure from the standard model used in Economics, we apply inequality criteria to allocation problems with indivisible goods, i.e., to the kind of problem typically analysed in Multiagent Systems. This gives rise to the combinatorial optimisation problem of computing an allocation that reduces inequality with respect to an initial allocation (and the closely related problem of minimising inequality), for a chosen inequality measure. We define this problem, we discuss the computational complexity of various aspects of it, and we formulate a generic approach to designing modular algorithms for solving it using integer programming.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*; J.4 [**Social and Behavioral Sciences**]: Economics

General Terms

Theory, Economics

Keywords

Fair Division, Inequality, Computational Complexity

1. INTRODUCTION

Many applications studied in Artificial Intelligence involve the design of mechanisms for dividing resources amongst a group of agents. One important criterion for assessing the fairness of a mechanism is the level of economic equality it can ensure. If it produces outcomes under which every agent experiences the same level of utility, then we speak of perfect equality. In all other cases, allocations exhibit some level of *inequality*, which raises the question of how to *measure* inequality and then how to *reduce* or *minimise* it.

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.*

Inequality criteria have been widely used in the social sciences, ranging from Economics [32], to Geography [23], to Public Health [17]. In contrast to this, even though some recent work in Multiagent Systems (as well as Artificial Intelligence and Electronic Commerce) has emphasised the importance of economic fairness (see, e.g., [6, 12, 22]), concepts specifically pertaining to inequality have largely been neglected in these disciplines. As argued in detail by Chevaleyre et al. [8], given the broad spectrum of agent-based applications, we need to be able to use different types of system objectives, including those that make explicit reference to fairness. For instance, a user may only agree to let a software agent negotiate on her behalf in a given system, if the designer of that system can provide guarantees regarding bounds on inequality the participating agents may be subjected to. Here we want to take a first step towards closing this gap by developing the foundations of inequality reduction in a way that is relevant to Multiagent Systems, focussing on computational and representational issues.

While the theoretical literature on inequality has dealt with axiomatic characterisations and the practical literature has calculated and interpreted inequality indices for various types of data, here we formulate *inequality reduction as a combinatorial optimisation problem:* Given a set of goods and the preferences of a group agents, how can we compute an allocation of goods to agents that reduces, or even minimises, inequality? We formulate the relevant problems, discuss their complexity, and provide algorithms for solving them. Along the way, we pay special attention to the choice of language for representing agent preferences.

In Section 2 we introduce our model of fair division, including fairness criteria and preference representation languages. In terms of fairness criteria, we cover the *Pigou-Dalton principle*, the *Lorenz curve*, and the most important *inequality indices*, namely the *Gini index*, the *Robin Hood index*, and the family of *Atkinson indices*. In terms of languages, we use the *XOR-language* and the *OR-language* familiar from combinatorial auctions, as well as languages based on *weighted propositional formulas*. Our complexity results are given in Section 3, while Section 4 shows how to formlate modular algorithms based on integer programming. Section 5 discusses related work and Section 6 concludes.

2. THE MODEL

In this section we introduce the formal model for fair division we shall be working with, drawing on both the literature on measuring economic inequality [2, 36, 25] and the literature on preference representation (see, e.g., [16]).

2.1 Allocation of Indivisible Goods

Let $\mathcal{N} = \{1, \ldots, n\}$ be a finite set of *agents* and let $\mathcal{G}$ be a finite set of *goods*. An *allocation* is a function $A : \mathcal{N} \to 2^{\mathcal{G}}$ with $A(i) \cap A(j) = \emptyset$ for any $i \neq j$, mapping agents to the bundles of goods they receive. That is, the goods are indivisible (each good is to be allocated to a single agent in its entirety) and the use of a good cannot be shared amongst more than one agent. If $A(1) \cup \cdots \cup A(n) = \mathcal{N}$, then A is called a *complete* allocation; otherwise it is called a *partial* allocation. Below, we shall use the letters A, A' for complete and P for partial allocations. When an allocation is not specifically referred to as being partial, then it is understood to be complete. We say that the allocation A *extends* the partial allocation P if $A(i) \supseteq P(i)$ for all agents $i \in \mathcal{N}$.

Each agent $i \in \mathcal{N}$ is equipped with a *utility function* $u_i : 2^{\mathcal{G}} \to \mathbb{Q}$, mapping any possible bundle she might receive to the value she assigns to it. We shall restrict attention to utility functions u that are *normalised* (that is, $u(\emptyset) = 0$) and *monotonic* (that is, $X \subseteq Y$ implies $u(X) \leqslant u(Y)$). A utility function u is called *additive* in case $u(X) = \sum_{x \in X} u(\{x\})$ for every bundle $X \subseteq \mathcal{G}$—note that this is an assumption that we usually will *not* make.

We use $u_i(A)$ as a shorthand for $u_i(A(i))$. Every allocation A induces a *utility vector* $\boldsymbol{u}(A) = (u_1(A), \ldots, u_n(A)) \in \mathbb{Q}^n$ as well as an *ordered utility vector* $\boldsymbol{u}^*(A) \in \mathbb{Q}^n$, which is obtained from $\boldsymbol{u}(A)$ by rearranging its elements in ascending order. That is, while $u_1(A)$ is the utility experienced by the *first* agent under allocation A, $u_1^*(A)$ is the utility of the *poorest* agent for the same allocation.

A *scenario* is a triple $\langle \mathcal{N}, \mathcal{G}, \mathcal{U} \rangle$, where $\mathcal{N}$ is a set of agents, $\mathcal{G}$ is a set of goods, and $\mathcal{U}$ is a vector of utility functions over subsets of $\mathcal{G}$, one for each agent in $\mathcal{N}$.

The model described here is widely used in Computer Science, e.g., for combinatorial auctions [9] and other forms of multiagent resource allocation [8], except that there is no monetary component here. An important difference between this model and the model used in the Welfare Economics literature, which deals with income distributions rather than the allocation of indivisible goods [25], is that in our case the space of feasible utility vectors is "sparse". For example, given an allocation that induces the utility vector $(3, 32, 4)$, we cannot assume that it will necessarily be possible to find an allocation with utilities $(13, 13, 13)$. In other words, utility is not transferable. As we shall see, this means that some of the results from the literature do not apply to our model.

2.2 Measuring Inequality

We now want to compare alternative allocations in terms of the inequality exhibited by the utility vectors they generate. For example, which should we rate as being more equal, $(1, 2, 8, 9)$ or $(1, 3, 5, 11)$? When there are just two agents, it is clear what inequality reduction means: the difference in utility between the two agents should reduce. If we combine this with a weak efficiency requirement, namely that average utility should not be diminished in the process, then we obtain the concept of a Pigou-Dalton transfer [10, 30].

Definition 1. *A move from allocation A to allocation A' is called a* ***Pigou-Dalton transfer*** *if there are two agents $i, j \in \mathcal{N}$ such that the following conditions are satisfied:*

(i) $A(k) = A'(k)$ *for all* $k \in \mathcal{N} \setminus \{i, j\}$;
(ii) $|u_i(A) - u_j(A)| > |u_i(A') - u_j(A')|$;
(iii) $u_i(A) + u_j(A) \leqslant u_i(A') + u_j(A')$.

That is, only i and j are affected, inequality is reduced, and total utility is not diminished. The *Pigou-Dalton principle* says that any Pigou-Dalton transfer should be considered as reducing (or at least not increasing) inequality.[1]

A useful representation of the distribution of wealth in a given society is the *Lorenz curve* [24]. It charts k against the cumulative utility of the k poorest agents.

Definition 2. *For any allocation A and any agent $k \in \mathcal{N}$, let $L_k(A) := \sum_{i=1}^{k} u_i^*(A)$. Then the vector $(L_1(A), \ldots, L_n(A))$ is called the* ***Lorenz curve*** *of A.*

That is, $L_k(A)$ is the sum of the utilities of the k poorest agents under A. $L_1(A)$ is known as the *egalitarian social welfare* of A and $L_n(A)$ is the *utilitarian social welfare* of A [25]. The *mean utility* of A is $\mu(A) := L_n(A)/n$. Intuitively, the closer the Lorenz curve is to the *line of perfect equality* $k \mapsto k \cdot \mu(A)$, the less inequality there is in society. We write $L(A) < L(A')$ if $L_k(A) \leqslant L_k(A')$ for all agents $k \in \mathcal{N}$ and that inequality is strict in at least one case. If $L(A) < L(A')$, then allocation A is said to be *Lorenz dominated* by A'; and a move from A to A' is called a *Lorenz improvement*. An allocation that is not Lorenz dominated by any other allocation is called *Lorenz optimal*.

Analogously, allocation A is said to be *Pareto dominated* by A' if no agent is worse off in A' and at least one of them is better off (i.e., if $u_i(A) \leqslant u_i(A')$ for all agents i and $u_i(A) < u_i(A')$ for at least one of them). *Pareto improvements* and *Pareto optimal* allocations are defined accordingly. Pareto optimality encodes a very basic efficiency requirement.

Clearly, Pigou-Dalton transfers and Pareto improvements are also Lorenz improvements. In the setting with transferable utility, furthermore every Lorenz improvement can be implemented as a sequence of Pigou-Dalton transfers and Pareto improvements [25, Lemma 2.3] While the same is not true in the context of allocating indivisible goods [12, Section 6.2], this connection nevertheless underlines the significance of the Lorenz curve for judging the quality of allocations when both inequality aversion and efficiency are at stake. Yet, Lorenz domination only offers a partial ranking of allocations; for allocations with intersecting Lorenz curves we need additional criteria to differentiate between them.

An *inequality index* is a function mapping allocations (or, equivalently: utility vectors) to the interval $[0, 1]$, with 0 representing perfect equality and 1 representing complete inequality. The *Gini index* is the most widely used inequality index [14]. It is defined as the ratio of (1) the area between the line of perfect equality and the Lorenz curve and (2) the full area below the line of perfect equality.

Definition 3. *The* ***Gini index*** *of allocation A is defined as follows:*

$$G(A) := \frac{\sum_{k=1}^{n} k \cdot \mu(A) - L_k(A)}{\sum_{k=1}^{n} k \cdot \mu(A)}$$

Another widely used inequality index is the *maximum relative mean deviation*, also known as the *Robin Hood index*. It is attractive due to its simplicity and has been re-invented a number of times [18]. It is defined as the ratio of (1) the maximum distance between the line of perfect equality and the Lorenz curve and (2) the mean utility.

[1] In the standard definition, condition *(iii)* requires *equality* of total utility [25]. Our definition maintains the spirit of the original Pigou-Dalton principle and is more appropriate for a framework without transferable utility.

DEFINITION 4. *The **Robin Hood index** of allocation A is defined as follows:*

$$H(A) \quad := \quad \frac{1}{\mu(A)} \cdot \max\{k \cdot \mu(A) - L_k(A) \mid k \in \mathcal{N}\}$$

Another interpretation of the Robin Hood index is this: Call an agent *rich* if her utility exceeds the mean utility, and *poor* otherwise. Then the Robin Hood index is equal to the proportion of total utility we need to move from rich to poor agents to make the two groups equally well off on average.

While the Gini and the Robin Hood index are widely used and based on plausible geometric properties of the Lorenz curve, they do not directly reflect any normative value judgements. Atkinson [2] pioneered the design of inequality indices in terms of social welfare orderings, for which axiomatisations reflecting such normative judgments are available. A *social welfare ordering* (SWO) is a complete order declared on the space of utility vectors, and thereby also on the set of allocations [25]. Examples include the utilitarian SWO (ranking allocations in terms of the sum of utilities they generate), the egalitarian SWO (ranking them in terms of minimum utility), and the Nash SWO (ranking them in terms of the product of utilities).

Let R be a SWO. Then for every allocation A there exists a value $\tilde{u}_R(A) \in \mathbb{Q}$ such that R is indifferent between the vectors $(\tilde{u}_R(A), \ldots, \tilde{u}_R(A))$ and $(u_1(A), \ldots, u_n(A))$. That is, if we want to create a situation where all agents enjoy identical levels of utility and that is socially as desirable as A, then we have to give utility $\tilde{u}_R(A)$ to each agent.

DEFINITION 5. *The **Atkinson index** (based on the social welfare ordering R) of allocation A is defined as follows:*

$$I_R(A) \quad := \quad 1 - \frac{\tilde{u}_R(A)}{\mu(A)}$$

Under some technical restrictions (see, e.g., [36]), we can assume $0 \leqslant \tilde{u}_R(A) \leqslant \mu(A)$, i.e., I_R is a well-formed inequality index. For example, for the Nash SWO we obtain $I_{\text{Nash}}(A) = 1 - \frac{\sqrt[n]{\prod_{i \in \mathcal{N}} u_i(A)}}{\mu(A)}$ and for the egalitarian SWO we get $I_{\text{egal}}(A) = 1 - \frac{\min_{i \in \mathcal{N}} u_i(A)}{\mu(A)}$. The utilitarian SWO is not a good choice if we are interested in measuring inequality, as we get $I_{\text{util}}(A) = 1 - \frac{\mu(A)}{\mu(A)} = 0$ for any A. That is, the utilitarian SWO is completely insensitive to inequality.

2.3 Preference Representation Languages

Before we can start thinking about algorithms for reducing inequality we have to decide how to represent the utility functions of the agents. So far we have only said that a utility function is a function $u_i : 2^{\mathcal{G}} \to \mathbb{Q}$. We could represent u_i *explicitly* as a list of pairs of bundles $B \subseteq \mathcal{G}$ and values $u_i(B)$, but that would be highly wasteful.

Instead we will work with a number of well-known *preference representation languages* (see, e.g., [8, 26, 39]):

- *XOR-language:* An *atomic bid* is a pair $\langle S, w \rangle$, where $S \subseteq \mathcal{G}$ and $w \in \mathbb{Q}^+$. An XOR-bid is an expression of the form $\langle S_1, w_1 \rangle \text{ XOR } \cdots \text{ XOR } \langle S_m, w_m \rangle$. It defines the utility function $u : X \mapsto \max\{w_i \mid X \supseteq S_i\}$.
- *OR-language:* An OR-bid is a combination of atomic bids $\langle S_1, w_1 \rangle \text{ OR } \cdots \text{ OR } \langle S_m, w_m \rangle$. It defines the function $u : X \mapsto \max_{X_1 \uplus \cdots \uplus X_m \subseteq X} \sum \{\!\{ w_i \mid X_i \supseteq S_i \}\!\}$.[2]
- Identify $\mathcal{G}$ with a set of propositional variables. A *goalbase* $G = \{(\varphi_i, w_i)\}_i$ is a set of *weighted goals*, where φ_i is a propositional formula over $\mathcal{G}$ and $w_i \in \mathbb{Q}$. G induces the utility function $u : X \mapsto \sum \{\!\{ w_i \mid X \models \varphi_i \}\!\}$, where $X \models \varphi_i$ means that φ_i is true in the model corresponding to the set X (i.e., variable p is true if $p \in X$). Various restrictions are possible: e.g., formulas φ_i might be restricted to literals or clauses, and weights w_i to positive numbers.

The XOR-language was introduced by Sandholm [34] in the context of modelling bids in combinatorial auctions. It can express all normalised monotonic utility functions, and only those [26, Proposition 9.2]. On the downside, it is only slightly more compact than the explicit representation (the only advantage is the built-in monotonicity assumption). The OR-language is the "traditional" language for bidding in combinatorial auctions. It is not fully expressive, but can only encode utility functions *without substitutatbilities* [26, Proposition 9.1]. The language of weighted goals was introduced by Pinkas [31], under the name of *penalty logic*. For combinatorial auctions, it has been argued to be superior to XOR/OR-type languages by Boutilier and Hoos [5] and it has been used extensively in computational studies of fair division (see, e.g., [6]). The full language can express all utility functions (see, e.g., [39, Corollary 3.7]) and the language of non-tautological negation-free formulas with positive weights expresses precisely the class of normalised monotonic utility functions [39, Theorem 3.11].

The most basic task concerning a language for representing utility functions is to compute the value of the represented function for a given bundle of goods.

DEFINITION 6. *Given the representation of utility function u in language $\mathcal{L}$, a bundle $B \subseteq \mathcal{G}$, and a bound $K \in \mathbb{Z}$, the* EVALUTIL *problem asks whether $u(B) \geqslant K$.*

For most languages, EVALUTIL is (and should be!) easy.

FACT 1. EVALUTIL *can be decided in polynomial time for the XOR-language and any weighted goal language.*

PROOF. Immediate: for the XOR-language this is a simple look-up; for weighted goals we need to solve a linear number of model checking problems for propositional logic. □

The OR-language is unusual in the sense that already this most basic task is intractable.

FACT 2. EVALUTIL *is NP-complete for the OR-language.*

PROOF. By reduction from SET PACKING [13].[3] □

3. COMPLEXITY RESULTS

In this section we define two decision problems that are relevant to the broader goal of computing allocations with low levels of inequality, and we analyse their complexity.

[2] We use $X \uplus Y$ to denote the disjoint union of two sets and $\{\!\{\cdots\}\!\}$ for multisets.

[3] It is well-known that the *winner determination problem* for combinatorial auctions is NP-hard under the OR-language [33], and it is easy to see that the standard reduction applies even when there is just a single bidder—in which case the winner determination problem is equivalent to EVALUTIL.

3.1 Pigou-Dalton Improvements

We first discuss the problem of deciding whether a given allocation admits a Pigou-Dalton transfer.

DEFINITION 7. *Given a scenario $\langle \mathcal{N}, \mathcal{G}, \mathcal{U} \rangle$, an allocation A, and a partial allocation P, the* PIGOU-DALTON IMPROVEMENT *problem asks whether there exists an allocation A' extending P such that (A, A') is a Pigou-Dalton transfer.*

The reason for including P in Definition 7 is that this ensures that the *search problem* of *computing* A' can be reduced to a polynomial number of instances of the *decision problem* about the *existence* of A' formulated above.[4] Hence, any complexity result concerning PIGOU-DALTON IMPROVEMENT will be directly relevant to the corresponding search problem (which is what interests us in practice).

The complexity of PIGOU-DALTON IMPROVEMENT depends on the language used to represent utility functions.

PROPOSITION 3. PIGOU-DALTON IMPROVEMENT *can be decided in polynomial time when the XOR-language is used.*

PROOF. An exhaustive search, which is polynomial for the XOR-language, will produce the desired result: Let A be the current allocation and P the partial allocation we are asked to extend. For a particular pair of agents, i and j, we can check whether a Pigou-Dalton transfer is possible by going through the list of atomic bids for i and j and, for each pair of atomic bids S_i, S_j, checking whether (1) $S_i \cap S_j = \emptyset$, (2) $S_i, S_j \subseteq A(i) \cup A(j)$, (3) $S_i \supseteq P(i)$ and $S_j \supseteq P(j)$, and (4) their weights satisfy the Pigou-Dalton conditions. This can clearly be done in polynomial time. As the number of pairs of agents is quadratic, the problem is in P. □

This simple algorithm will not work for compact representation languages. For compact languages, we usually cannot just go through all *relevant* bundles. Going through *all* bundles would take an exponential amount of time. We now show that we cannot do much better than that, for any of the compact representation languages we have considered.

PROPOSITION 4. PIGOU-DALTON IMPROVEMENT *is NP-hard for the OR-language.*

PROOF. By reduction from EVALUTIL for the OR-language. Let $\texttt{Bid}_u$ be an OR-bid representing utility function u. To check whether there exists a bundle B with $u(B) \geqslant K$ (i.e., whether $u(\mathcal{G}) \geqslant K$ for the full set of goods $\mathcal{G}$), construct the following two-agent instance of PIGOU-DALTON IMPROVEMENT. Let Ω be a number that is greater than the sum of all weights in $\texttt{Bid}_u$, let ϵ be a number than is lower than any of those weights, and let x be a new good not in $\mathcal{G}$. Agent 1 has the OR-bid $\texttt{Bid}_u$ OR $\langle \{x\}, \Omega \rangle$. Agent 2 has the OR-bid $\langle \mathcal{G}, K-\epsilon \rangle$ OR $\langle \{x\}, \Omega \rangle$. Let A_0 be the allocation that assigns x to agent 1 and all of $\mathcal{G}$ to agent 2. The utility vector of A_0 is $(\Omega, K-\epsilon)$. Clearly, the only chance of finding a Pigou-Dalton transfer is to give item x to agent 2 and all of $\mathcal{G}$ to agent 1, resulting in an allocation with the utility vector $(u(\mathcal{G}), \Omega)$. Hence, there exists a Pigou-Dalton transfer *iff* $u(\mathcal{G}) \geqslant K$. □

[4]By a standard argument [29, Example 10.3], we can compute A' using $O(n \cdot |\mathcal{G}|)$ calls to a PIGOU-DALTON IMPROVEMENT oracle by instantiating the allocation P item-by-item.

In view of Fact 2, this is not a surprising result. We stress that Proposition 4 only establishes NP-hardness, not NP-completeness. We do not know whether the problem is in NP and it is in fact conceivable that it might not be. The reason is—roughly speaking—that verifying whether a move from A to A' is a Pigou-Dalton transfer not only requires checking that A' has *at least* a certain quality, but also that A has *at most* a certain quality.[5]

What about other languages? Take the weighted goal language where all formulas are required to be *atoms*. This is a very limited language, which can only express *additive* utility functions [39, Corollary 3.8]. Consider a scenario with just two agents who have the same utility function (represented by the same set of weighted atoms) and an allocation A allocating all goods to agent 1. Now suppose we do not just want to know whether there exists a Pigou-Dalton improvement over A, but whether there exists one that reduces inequality to below a given threshold K. In other words, we are asking: given a vector of numbers $(w_1, \ldots, w_m) \in \mathbb{N}^m$, is there an index set $S \subseteq \{1, \ldots, m\}$ such that $|\sum_{i \in S} w_i - \sum_{i \notin S} w_i| < K$? But this is the well-known NP-hard PARTITION problem [13]. Hence, deciding PIGOU-DALTON IMPROVEMENT *with quality guarantees* (in the above sense) is NP-hard. This intractability result immediately extends to more expressive weighted goal languages as well.

Whether also the plain PIGOU DALTON IMPROVEMENT problem is NP-hard for the language of weighted atoms is an open question. It might seem immediately obvious that it should be NP-hard (so let us see why it might not be). A natural attempt at an NP-hardness proof would make use of the following variant of the PARTITION problem, which we call BETTER PARTITION: we are given a vector $(w_1, \ldots, w_m) \in \mathbb{N}^m$ and an index set $S \subseteq \{1, \ldots, m\}$ and ask whether there exists a "better" index set $S' \subseteq \{1, \ldots, m\}$ such that $|\sum_{i \in S'} w_i - \sum_{i \notin S'} w_i| < |\sum_{i \in S} w_i - \sum_{i \notin S} w_i|$. There is no immediate reduction from PARTITION to BETTER PARTITION: For instance, if $K < 1$ (i.e., when we are looking for a perfect partition), then we might have to solve an exponential number of BETTER PARTITION instances before reaching that perfect partition (if each step only reduces the difference by 1 and the sum of weights is exponential in the size of the problem encoding). Also the fact that PARTITION can be ϵ-approximated in polynomial time for any arbitrarily small ϵ (see, e.g., [29, Theorem 13.5]) does not help, because there might still be an exponential number of improvement steps between the solution found by an approximation algorithm and the optimum. An intuitive reason why BETTER PARTITION *might* be tractable is this: if S corresponds to a very uneven partition, then it will be easy to improve on it and to find S'; while if S already is a very good partition, then it is at least conceivable that a smart algorithm could exploit the information that S represents a high-quality approximation (note that the K in PARTITION does not provide any such helpful information).

[5]This combination of positive and negative requirements is reminiscent of DP, the complexity class of languages that are the intersection of a language in NP and a language in coNP [29]. This suggests that PIGOU-DALTON IMPROVEMENT for the OR-language might be DP-hard (though not necessarily in DP). What we can say with certainty is that the problem is in Δ_2^p, the class of problems that can be solved with a polynomial number of calls to an NP-oracle [29].

We are able to prove the following intriguing result:[6]

PROPOSITION 5. PIGOU-DALTON IMPROVEMENT *cannot be decided in polynomial time when the language of weighted atoms is used, unless NP = coNP.*

PROOF. In case there are exactly two agents with the same additive utility function, our problem is equivalent to the BETTER PARTITION problem defined above. So let us show that BETTER PARTITION cannot be decided in polynomial time, unless NP = coNP. First, observe that the problem NO PERFECT PARTITION, i.e., the problem of deciding whether a given set of numbers can*not* be partitioned into two subsets with equal sums, is coNP-hard.

Now, for the sake of contradiction, suppose there exists a polynomial algorithm `Alg` that replies YES *iff* a given partition can be improved upon. We can use `Alg` to prove that NO PERFECT PARTITION must be in NP: We need to show that when someone claims that the answer to a given NO PERFECT PARTITION instance should be YES and provides a suitable certificate, then we can verify the correctness of that certificate in polynomial time. A suitable certificate would be a partition of the set of numbers into two subsets that minimises the difference of their sums. To verify correctness (i.e., to verify that indeed no perfect partition exists), we first check that the difference in sums is indeed non-zero (this is easy) and then that it cannot be improved upon (which is exactly the problem `Alg` can decide in polynomial time). Hence, NO PERFECT PARTITION is in NP.

In summary, we have shown that there exists a coNP-hard problem (namely NO PERFECT PARTITION) that is in NP. But this means that coNP $\subseteq$ NP. This in turn would immediately imply NP = coNP, so we are done. □

Note that Proposition 5 does *not* imply that PIGOU-DALTON IMPROVEMENT for the language of weighted atoms is NP-hard (but the problem is easily seen to be in NP). So this really is a problem that very much sits at the borderline of the tractable and the intractable.

3.2 Lorenz Improvements

We now turn to computing Lorenz improvements. The corresponding decision problem is defined as follows.

DEFINITION 8. *Given a scenario $\langle \mathcal{N}, \mathcal{G}, \mathcal{U} \rangle$, an allocation A, and a partial allocation P, the* LORENZ IMPROVEMENT *problem asks whether there exists an allocation A' extending P such that $L(A) < L(A')$.*

Unsurprisingly, this is intractable for the OR-language:

PROPOSITION 6. LORENZ IMPROVEMENT *is NP-complete for the OR-language.*

PROOF. NP-hardness is proved by reduction from SET PACKING, similarly to Fact 2. We omit the details for lack of space. We postpone arguing for NP-membership (which, in view of Footnote 5, is not obvious) to Section 4.4. □

We now also get an intractability result for XOR:

PROPOSITION 7. LORENZ IMPROVEMENT *is NP-complete for the XOR-language.*

PROOF. NP-membership is immediate. We show NP-hardness by a reduction from the problem of deciding whether, for a given K, there exists an allocation with utilitarian social welfare $\geqslant K$, which is known to be NP-hard for the XOR-language [19, Theorem 12.1]. Suppose we are given an instance of the latter problem. Now add one additional agent to the problem, with a single atomic bid $\langle \mathcal{G}, K-\epsilon \rangle$, where ϵ is smaller than any of the differences we can construct from the weights used in the bids of the original agents (e.g., if all original weights are integers, then we can set $\epsilon := 0.5$). The question of whether an allocation with social welfare $\geqslant K$ exists is not affected by this addition.

Let A be the allocation where all goods are given to the new agent. As all other agents have utility 0 in A, the social welfare of A is $L_n(A) = K-\epsilon$. The crucial insight now is the fact that, for any allocation A', we have $L(A) < L(A')$ if and only if A' has strictly higher social welfare than A, i.e., if $K \leqslant L_n(A')$. This completes the reduction. □

Our proof of NP-hardness above is very general. It extends to any preference representation language $\mathcal{L}$ for which utilitarian social welfare optimisation is NP-hard—and remains hard when we assume that all agents have normalised utility functions. This is the case for many weighted goal languages [38, p. 121]. We state this result explicitly for a few particularly important languages:[7]

PROPOSITION 8. LORENZ IMPROVEMENT *is NP-complete for any of the following languages: the language of positively weighted 2-cubes; the language of positively weighted 2-clauses; the language of negation-free 2-cubes; the language of negation-free 2-clauses; and the language of positively weighted non-tautological negation-free 3-cubes.*

PROOF. NP-hardness follows from results by Uckelman [38] on NP-hardness of utilitarian social welfare optimisation for these languages together with the proof technique used to establish Proposition 7. NP-membership is immediate for any weighted goal language (by Fact 1). □

For weighted atoms, on the other hand, utilitarian social welfare optimisation is easily seen to be polynomial (simply allocate each good to the agent assigning the highest weight to it), i.e., our technique does not apply here.

Finally, observe that for any language for which LORENZ IMPROVEMENT is NP-hard the corresponding problem LORENZ OPTIMALITY, i.e., the problem of deciding whether a given allocation is Lorenz optimal, must be coNP-hard (because deciding Lorenz optimality amounts to deciding whether there exists *no* Lorenz improvement).

4. COMPUTING FAIR ALLOCATIONS

In this section we show how to formulate the most important problems of inequality reduction in combinatorial domains in the language of *integer programming* [35].[8] That this is possible in principle is clear, as every problem in NP can be embedded into integer programming [28]. Our contribution is to show how to do this in a simple and modular manner that allows us to easily change either the inequality measure

[6] I'm indebted to Harry Buhrman, Bruno Loff and Leen Torenvliet for the main insight at the heart of this result; see also their work on approximations of knapsack problems [7].

[7] Recall that a k-clause is a disjunction of at most k literals and a k-cube is a conjunction of at most k literals.

[8] We shall assume familiarity with integer programming (for an introduction see, e.g., [37]).

employed or the preference representation language used. Providing an integer programming formulation means that powerful off-the-shelf tools can be used to solve medium-size instances of these problems in practice—although further optimisation will certainly be required in the future.

We start out by specifying constraints that allow us to implement a solution to the LORENZ IMPROVEMENT problem, and then show how to combine this with objective functions based on inequality indices. For ease of presentation, we present algorithms for the XOR-language first, and then briefly comment on extensions to other languages.

While inequality reduction for the allocation of indivisible goods with compactly represented preferences has not been discussed in the literature before, we are able to rely on familiar techniques from work on inequality minimisation for additive utilities (see, e.g., [27]) and welfare optimisation in combinatorial domains, particularly work on combinatorial auctions (see, e.g., [1, 4]).

4.1 Lorenz Improvements, XOR-language

Suppose all agents express their utility functions using the XOR-language, and suppose the current allocation is A. We want to define a set of linear inequalities that constrain a possible follow-up allocation A' to be a Lorenz improvement over A. We will associate A with a vector of binary decision variables $\boldsymbol{x}$ (already instantiated, as A is given) and we will associate A' with another vector of binary decision variables $\boldsymbol{y}$ (to be instantiated). If a partial allocation P fixing some of the assignments of goods in A' is already given, then we instantiate the corresponding elements of $\boldsymbol{y}$ accordingly. For expository convenience, assume each agent has supplied the same number of atomic bids m.

Let $\langle S_{ij}, w_{ij}\rangle$ be the jth atomic bid of agent i. Introduce a vector $\boldsymbol{y}$ of binary decision variables consisting of three types of variables: $y_{ij} \in \{0,1\}$ with $y_{ij} = 1$ if and only if the jth atomic bid of agent i is accepted under A'; $y_i^k \in \{0,1\}$ with $y_i^k = 1$ if and only if agent i is on position k in the ordered utility vector for A'; and $y_{ij}^k \in \{0,1\}$ with $y_{ij}^k = 1$ if and only if both $y_{ij} = 1$ and $y_i^k = 1$. That is, $\boldsymbol{y}$ determines allocation A', and it also encodes the poor-to-rich ordering on the agents under that allocation.

The following constraints ensure the instantiation of y_{ij}^k is consistent with the choices made for y_{ij} and y_i^k.

$$(\forall i, k \leqslant n, \forall j \leqslant m) \qquad y_{ij}^k \leqslant y_{ij} \tag{1}$$

$$(\forall i, k \leqslant n, \forall j \leqslant m) \qquad y_{ij}^k \leqslant y_i^k \tag{2}$$

$$(\forall i, k \leqslant n, \forall j \leqslant m) \qquad y_{ij}^k \geqslant y_{ij} + y_i^k - 1 \tag{3}$$

The next constraint comes from the XOR-language: at most one atomic bid per agent can be accepted.

$$(\forall i \leqslant n) \qquad \sum_{j \leqslant m} y_{ij} \leqslant 1 \tag{4}$$

To ensure the poor-to-rich ordering is a proper ordering we stipulate that for every position k there is exactly one agent i and that for every agent i there is exactly one position k.

$$(\forall k \leqslant n) \qquad \sum_{i \leqslant n} y_i^k = 1 \tag{5}$$

$$(\forall i \leqslant n) \qquad \sum_{k \leqslant n} y_i^k = 1 \tag{6}$$

To improve readability, we define a "macro" to refer to elements of the ordered utility vector corresponding to $\boldsymbol{y}$:

$$(\forall k \leqslant n) \qquad u_k^*(\boldsymbol{y}) := \sum_{i \leqslant n, j \leqslant m} y_{ij}^k \cdot w_{ij}$$

Now we can formulate a constraint that ensures that the poor-to-rich ordering is consistent with actual utilities.

$$(\forall k \leqslant n-1) \qquad u_k^*(\boldsymbol{y}) \leqslant u_{k+1}^*(\boldsymbol{y}) \tag{7}$$

Here is a second macro, which we use to refer to elements of the Lorenz curve for the allocation corresponding to $\boldsymbol{y}$:

$$(\forall k \leqslant n) \qquad L_k(\boldsymbol{y}) := \sum_{\ell \leqslant k} u_\ell^*(\boldsymbol{y})$$

$L_k(\boldsymbol{x})$ is defined analogously. So far, we have ensured that $\boldsymbol{y}$ represents a feasible allocation together with a correct ordering of the agents. Now suppose that the other set of binary decision variables, $\boldsymbol{x}$, has been instantiated so as to correctly represent A and the ordered utility vector for A. Here are the constraints for checking $L(A) < L(A')$:

$$(\forall k \leqslant n) \qquad L_k(\boldsymbol{x}) \leqslant L_k(\boldsymbol{y}) \tag{8}$$

$$\sum_{k \leqslant n} L_k(\boldsymbol{x}) < \sum_{k \leqslant n} L_k(\boldsymbol{y}) \tag{9}$$

This completely describes the problem LORENZ IMPROVEMENT. Note that there is no objective function to be maximised here. If we are only interested in finding *some* Lorenz improvement, we can simply "maximise" a constant function, subject to constraints (1)–(9).

4.2 Objective Functions, XOR-language

Given our program for Lorenz improvements, we can choose any objective function to maximise some quantity over $\boldsymbol{y}$ to search for the "best" Lorenz improvement within the set of feasible improvements. For instance, we may wish to maximise the sum of utilities, subject to the chosen allocation ($\boldsymbol{y}$) representing a Lorenz improvement over the *status quo* ($\boldsymbol{x}$). This amounts to making $L_n(\boldsymbol{y})$ the objective function, which is equivalent to the *winner determination problem* in combinatorial auctions [9], for which integer programming implementations are available for the XOR-language [19].

Here we show instead how to compute the best allocation according to either the Gini index or the Robin Hood index, from the set of allocations that Lorenz dominate the *status quo*. Our approach is rather general and can be adapted to other inequality indices as well. Recall Definition 5, which shows how to translate an SWO into an inequality index. We can take the reverse approach. Take any inequality index I. Together with $\boldsymbol{u}$ (determining, amongst other things, μ), I induces a function $W_I : 2^{\mathcal{G}} \to \mathbb{Q}$:

$$W_I(A) = (1 - I(A)) \cdot \mu(A)$$

We can think of W_I as a *collective utility function* (CUF), mapping each allocation to a single value [25]. The corresponding SWO (ranking allocations in terms of those values) again induces the Atkinson index I. The better an allocation in terms of W_I, the less inequality it exhibits according to I, and *vice versa*. If we apply this technique to the Gini index, we obtain this CUF:

$$W_G(A) = \frac{2}{n(n+1)} \cdot \sum_{k=1}^{n} L_k(A)$$

That is, the higher $\sum_{k=1}^{n} L_k(A)$, the lower the Gini index of A. Now it is straightforward to implement optimisation wrt. the Gini index. Given an initial allocation, represented by $\boldsymbol{x}$, the following integer program computes an allocation with minimal Gini index from the set of allocations that Lorenz dominate $\boldsymbol{x}$, when the XOR-language is used:

```
maximise ∑_{k=1}^{n} L_k(y)
subject to constraints (1)-(9)
```

The solution is given by $\boldsymbol{y}$: if $y_{ij} = 1$, then give all the goods in S_{ij} to agent i.

While the CUF corresponding to the Gini index has been extensively discussed in the literature [3, 40], a CUF corresponding to the Robin Hood index appears not to have been formulated before. Applying the same approach as above, we obtain:

$$W_H(A) \quad = \quad \min\{L_k(A) - (k-1) \cdot \mu(A) \mid k \in \mathcal{N}\}$$

Again, the Atkinson index induced by W_H is precisely the Robin Hood index and optimising wrt. W_H means minimising inequality according to the Robin Hood index. Turning W_H into an objective function for an integer program is less immediate than for W_G. Here we give a solution based on *mixed integer programming*, making use of a non-integer variable z. The following set of constraints forces z to be at most as large as W_H for the allocation corresponding to $\boldsymbol{y}$:

$$(\forall k \leqslant n) \qquad z \quad \leqslant \quad L_k(\boldsymbol{y}) - (k-1) \cdot \frac{L_n(\boldsymbol{y})}{n} \qquad (10)$$

Now the following program computes an allocation with minimal Robin Hood index from the set of allocations that Lorenz dominate $\boldsymbol{x}$, when the XOR-language is used:

```
maximise z
subject to constraints (1)-(10)
```

Solutions are extracted as before.

4.3 Other Representation Languages

If we want to use a different preference representation language, we have to change constraint (4), encoding the XOR-condition, and we have to change the macro for computing $u_k^*(\boldsymbol{y})$. While we must omit the details for lack of space, we note that this is easily possible for both the OR-language and weighted goals. Indeed, for both languages, the winner determination problem for combinatorial auctions has been formulated as an integer program, and such an implementation necessitates both the encoding of the relevant language constraint and the definition of the utility experienced by any given agent under the solution allocation. For the OR-language this is standard [19]; for weighted goals we refer to Boutilier [4], who gives an integer program for a notational variant of the language of weighted goals used in this paper.

As a final remark, we note that the number of decision variables used in our implementation is linear in the size of the representation on agent preferences, albeit quadratic in the number of agents. The latter is due to the need of encoding the ordering of agents inherent in the definition of the Lorenz curve, and this appears impossible to avoid.

4.4 Relation to Complexity Results

Given that any problem expressible as an integer program (without an objective function) is in NP [28], the fact that we can encode Lorenz Improvement for the OR-language completes our proof of Proposition 6.

We do not know of a general method for encoding Pigou-Dalton Improvement in integer programming. The problematic issue is that the absolute-value function $|\cdot|$ (see Definition 1) is nonlinear. For any variant of Pigou-Dalton Improvement that is in NP there clearly are ways around this difficulty (because *every* problem in NP can be mapped into integer programming), but our discussion in Footnote 5 suggests hat this might be impossible for the OR-language.

5. RELATED WORK

In related work, Lesca and Perny [20] give (mixed) integer programming formulations for two fair division problems, one of which is that of finding an allocation that minimises the Gini index (the other is related to the Choquet integral). A crucial difference wrt. our model is that they assume utility functions to be *additive*, i.e., each agent assigns a value to each good and an agent's utility of a bundle is the sum of the values she assigns to the goods in that bundle. The intractability of the problems studied by Lesca and Perny stems from the fact that they impose *volume constraints* limiting the number of goods an agent might receive.

Also all other algorithmic work on inequality reduction we are aware of, such as the contributions of Ogryczak [27] and of Golden and Perny [15], make the same assumption of utility functions being *additive*. That is, in those works there is *no compact preference representation* involved, which is a defining feature of our approach, and which is of crucial significance for fair allocation problems with *indivisible* goods, which is the most widely studied scenario in Multiagent Systems [8]. Other related work discusses conditions under which negotiation in a multiagent system will converge to an allocation that is Lorenz optimal [12].

The complexity of computing fair allocations for other fairness criteria is analysed, amongst others, by Lipton et al. [22] and Bouveret and Lang [6]. In analogy to our results on the complexity of Lorenz Improvement, the problem of recognising Pareto improvements is known to be NP-hard for both the language of weighted negation-free k-cubes (also known as the *k-additive form*) and a language based on *straight-line programs*, while the corresponding problem of deciding Pareto optimality is coNP-hard [8, 11].

6. CONCLUSION

We have formulated the task of reducing (or minimising) economic inequality as a combinatorial optimisation problem when several indivisible goods need to get allocated to a group of agents. We have then discussed complexity-theoretic and algorithmic questions that arise in this setting. Particularly the conceptually simple notion of Pigou-Dalton transfer nicely demonstrates the complexity-theoretic subtleties involved when switching from one problem representation to another, an effect that is not typical for complexity results in multiagent resource allocation [8]. We also hope that our modular integer programming formulation will prove useful to those who wish to apply the theoretical concepts discussed here in practice.

Specific technical contributions aside, a declared aim of this paper is to bring the important topic of measuring economic inequality to the attention of a broader section of the research community in Multiagent Systems.

An interesting direction for future work would be to cover a wider spectrum of representation languages to complete the picture of complexity results presented here. It is also important to experiment with practical implementations of our algorithms. Relevant data sets on which to run those experiments would have to be developed in parallel with refining the algorithms. A possible starting point are existing approaches for generating high-quality test data for combinatorial auctions [21].

Acknowledgements. I would like to thank the audience of the 4th MARA Get-Together held in Paris in June 2010 for feedback on an early version of this paper and Leen Torenvliet for enlightening discussions on knapsack problems.

7. REFERENCES

[1] A. Andersson, M. Tenhunen, and F. Ygge. Integer programming for combinatorial auction winner determination. In *Proc. ICMAS-2000*, 2000.

[2] A. B. Atkinson. On the measurement of inequality. *Journal of Economic Theory*, 2:244–263, 1970.

[3] C. Blackorby and D. Donaldson. Measures of relative equality and their meaning in terms of social welfare. *Journal of Economic Theory*, 18(1):59–80, 1978.

[4] C. Boutilier. Solving concisely expressed combinatorial auction problems. In *Proc. AAAI-2002*, 2002.

[5] C. Boutilier and H. H. Hoos. Bidding languages for combinatorial auctions. In *Proc. IJCAI-2001*, 2001.

[6] S. Bouveret and J. Lang. Efficiency and envy-freeness in fair division of indivisible goods: Logical representation and complexity. *JAIR*, 32:525–564, 2008.

[7] H. Buhrman, B. Loff, and L. Torenvliet. Approximation algorithms and hardness of approximation for knapsack problems. Manuscript, CWI and University of Amsterdam, 2012.

[8] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J. A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica*, 30(1):3–31, 2006.

[9] P. Cramton, Y. Shoham, and R. Steinberg, editors. *Combinatorial Auctions*. MIT Press, 2006.

[10] H. Dalton. The measurement of the inequality of incomes. *The Economic Journal*, 30(119):348–361, 1920.

[11] P. E. Dunne, M. Wooldridge, and M. Laurence. The complexity of contract negotiation. *Artif. Intell.*, 164(1–2):23–46, 2005.

[12] U. Endriss, N. Maudet, F. Sadri, and F. Toni. Negotiating socially optimal allocations of resources. *JAIR*, 25:315–348, 2006.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, 1979.

[14] C. Gini. *Variabilitá e Mutabilitá*. C. Cuppini, Bologna, 1912.

[15] B. Golden and P. Perny. Infinite order Lorenz dominance for fair multiagent optimization. In *Proc. AAMAS-2010*, 2010.

[16] J. Goldsmith and U. Junker. Preference handling for artificial intelligence (editorial). *AI Magazine*, 29(4):9–12, 2008.

[17] B. P. Kennedy, I. Kawachi, and D. Prothrow-Stith. Income distribution and mortality: Cross sectional ecological study of the Robin Hood index in the United States. *BMJ*, 312:1004–1007, 1996.

[18] Y. Kondor. An old-new measure of income inequality. *Econometrica*, 39(6):1041–1042, 1971.

[19] D. Lehmann, R. Müller, and T. Sandholm. The winner determination problem. In Cramton et al. [9].

[20] J. Lesca and P. Perny. LP solvable models for multiagent fair allocation problems. In *Proc. ECAI-2010*, 2010.

[21] K. Leyton-Brown and Y. Shoham. A test suite for combinatorial auctions. In Cramton et al. [9].

[22] R. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *Proc. ACM EC-2004*, 2004.

[23] L. Long and A. Nucci. The Hoover index of population concentration: A correction and update. *The Professional Geographer*, 49(4):431–440, 1997.

[24] M. O. Lorenz. Methods of measuring the concentration of wealth. *Publications of the American Statistical Association*, 9(70):209–219, 1905.

[25] H. Moulin. *Axioms of Cooperative Decision Making*. Cambridge University Press, 1988.

[26] N. Nisan. Bidding languages for combinatorial auctions. In Cramton et al. [9].

[27] W. Ogryczak. Inequality measures and equitable approaches to location problems. *European Journal of Operational Research*, 122(2):374–391, 2000.

[28] C. H. Papadimitriou. On the complexity of integer programming. *JACM*, 28(4):765–768, 1981.

[29] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[30] A. C. Pigou. *Wealth and Welfare*. Macmillan, London, 1912.

[31] G. Pinkas. Reasoning, nonmonotonicity and learning in connectionist networks that capture propositional knowledge. *Artif. Intell.*, 77(2):203–247, 1995.

[32] M. Ravallion and S. Chen. Measuring pro-poor growth. *Economic Letters*, 78(1):93–99, 2003.

[33] M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally manageable combinational auctions. *Management Science*, 44(8):1131–1147, 1998.

[34] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artif. Intell.*, 135(1–2):1–54, 2002.

[35] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1986.

[36] A. Sen. *On Economic Inequality*. Oxford University Press, 1973.

[37] Y. Shoham and K. Leyton-Brown. *Multiagent Systems*. Cambridge University Press, 2009.

[38] J. Uckelman. *More than the Sum of its Parts: Compact Preference Representation over Combinatorial Domains*. PhD thesis, ILLC, University of Amsterdam, 2009.

[39] J. Uckelman, Y. Chevaleyre, U. Endriss, and J. Lang. Representing utility functions via weighted goals. *Mathematical Logic Quarterly*, 55(4):341–361, 2009.

[40] J. Weymark. Generalized Gini inequality indices. *Mathematical Social Sciences*, 1(4):409–430, 1981.