Complexity and Approximability of Social Welfare Optimization in Multiagent Resource Allocation[1]

Nhan-Tam Nguyen, Trung Thanh Nguyen, Magnus Roos, and Jörg Rothe

Abstract

A central task in multiagent resource allocation, which provides mechanisms to allocate (bundles of) resources to agents, is to maximize social welfare. We assume resources to be indivisible and nonshareable and agents to express their utilities over bundles of resources, where utilities can be represented in the bundle form, the k-additive form, and as straight-line programs. We study the computational complexity of social welfare optimization in multiagent resource allocation, where we consider utilitarian and egalitarian social welfare and social welfare by the Nash product. We prove that exact social welfare optimization by the Nash product is DP-complete for the bundle and the 3-additive form, where DP is the second level of the boolean hierarchy over NP. For utility functions represented as straight-line programs, we show NP-completeness for egalitarian social welfare optimization and social welfare optimization by the Nash product. Finally, we show that social welfare optimization by the Nash product in the 1-additive form is hard to approximate, yet we also give fully polynomial-time approximation schemes for egalitarian and Nash product social welfare optimization in the 1-additive form with a fixed number of agents.

1 Introduction

Multiagent resource allocation (MARA) deals with distributing resources to agents that have preferences over (bundles of) resources. These resources are assumed to be indivisible and nonshareable. Agents express their preferences by means of utility functions. Hence, every given allocation of resources to agents induces a vector of utilities that can be aggregated to a single value, the social welfare of this allocation. There are different notions of social welfare, ranging from the well-known utilitarian social welfare to egalitarian social welfare, to compromises between these two notions such as the Nash product and generalizations thereof (k-rank dictator functions, etc.).

In a bit more detail, *utilitarian social welfare* sums up the agents' individual utilities in a given allocation, thus providing a useful measure of the overall—and also of the average—benefit for society. For instance, in a combinatorial auction the auctioneer's aim is to maximize the auction's revenue (i.e., the sum of the prizes paid for the items auctioned), no matter which agent can realize which utility.

In contrast, *egalitarian social welfare* gives the utility of the agent who is worst off in a given allocation, which provides a useful measure of fairness in cases where the minimum needs of all agents are to be satisfied. For example, think of distributing humanitarian aid items (such as food, medical aid, blankets, tents, etc.) among the needy population in a disaster area (e.g., an area hit by an earthquake or a tsunami). Guaranteeing every survivor's continuing survival is the primary goal in such a scenario, and it is best captured by the notion of egalitarian social welfare.

As mentioned above, the *Nash product*, the product of the agents' utilities, can be seen as a compromise between these two approaches. On the one hand, it has the (strict) monotonicity property of utilitarian social welfare because an increase in any agent's utility leads to an increase of the Nash product (provided all agents have positive utility). On the other hand, the Nash product increases

[1]Preliminary versions of parts of this paper appear in the proceedings of the *11th International Joint Conference on Autonomous Agents and Multiagent Systems* [17], of the *6th European Starting AI Researcher Symposium* [16], and of the *12th International Symposium on Artificial Intelligence and Mathematics* [18]. This work was supported in part by DFG grant RO-1202/14-1, ARC grant DP110101792, a DAAD grant for a PPP project in the PROCOPE program, and a fellowship from the Vietnamese government.

as well when reducing inequitableness among agents by redistributing utilities, thereby providing a measure of fairness. Looking at the ordering that is induced by the allocations, the "social welfare ordering," Moulin [15] presents further beneficial properties of the Nash product. For example, the Nash product is uniquely characterized by independence of individual scale of utilities,[2] i.e., even if different "currencies" are used to measure the agents' utilities, the social welfare ordering remains unaffected.

All these notions of social welfare have in common that they seek to model that a high value of social welfare implies well-being among the society of agents (that is, for the group as a whole). Thus, the goal is to find allocations that maximize social welfare. How difficult is this task? The main purpose of this paper is to find answers to this question—for various central notions of social welfare, for distinct ways of representing utility functions, and for different ways of modeling MARA problems.

Although resource allocation problems are important for human agents as well, we are mostly concerned with (autonomous) software agents having individual utilities and acting in a shared environment (e.g., in a multiagent system). Therefore, it is of particular interest to study the computational complexity of MARA problems and to tackle computational hardness results by means of approximation algorithms. We present NP-completeness results for decision problems associated with egalitarian and Nash product social welfare optimization, and DP-completeness results for decision problems associated with Nash product social welfare optimization. We complement our results on the computational complexity of MARA decision problems by proving that the Nash product social welfare optimization problem is hard to approximate in the 1-additive form. For a fixed number of agents, we also give fully polynomial-time approximation schemes for egalitarian and of Nash product social welfare optimization in the 1-additive form.

This paper is organized as follows: In Section 2, we formalize the MARA framework that we have adopted from the profound survey by Chevaleyre et al. [5], and we introduce the needed background from complexity theory, including the perhaps lesser known complexity class DP, as well as some basic notions of approximation theory. Then, in Section 3, we briefly survey related work to see the context of our results. In Section 4 we present computational complexity results for the decision versions related to social welfare optimization, and in Section 5 we are concerned with approximability of social welfare optimization. In Section 6, we summarize our results and conclude with some open questions.

2 Preliminaries

2.1 Multiagent Resource Allocation Settings

We adopt the framework for multiagent resource allocation described in the survey by Chevaleyre et al. [5]. Let $A = \{a_1, a_2, \ldots, a_n\}$ be a set of n agents and let $R = \{r_1, r_2, \ldots, r_m\}$ be a set of m indivisible and nonshareable resources (i.e., each resource is assigned as a whole and can be assigned to only one agent). Subsets of R are called *bundles of resources*.

Every agent associates utility to every bundle of resources by specifying a utility function $u_i : 2^R \to \mathbb{F}$, where 2^R denotes the power set of R and $\mathbb{F}$ is a numerical set (such as the set $\mathbb{N}$ of nonnegative integers, the set $\mathbb{Z}$ of integers, the set $\mathbb{Q}$ of rational numbers, and the set $\mathbb{Q}^+$ of nonnegative rational numbers). The idea behind utility functions mapping *bundles* of resources rather than single resources to values in $\mathbb{F}$ is that agents might be willing to pay either more or less for a bundle than the sum of their utilities for this bundle's single items. For example, owning a pair of matching shoes is likely to be more valuable to an agent than the sum of the values each single shoe has for this agent. On the other hand, an agent who is willing to bid on 100 identical items might expect

[2] Similarly, utilitarian social welfare is characterized by independence of individual zeros of utilities: A constant shift of an agent's utility function does not change the social welfare ordering.

some discount and so has less utility for the bundle of 100 items than 100 times the utility assigned to a single item.

Let $U = \{u_1, u_2, \ldots, u_n\}$ be the set of the agents' *utility functions*. A triple (A, R, U) is called a *multiagent resource allocation setting* (a *MARA setting*, for short).

A concrete distribution of resources to agents is an allocation. Formally, for a given MARA setting (A, R, U), an *allocation* is a mapping

$$X : A \to 2^R$$

with $\bigcup_{a_i \in A} X(a_i) = R$ (i.e., every resource is given to some agent) and $X(a_i) \cap X(a_j) = \emptyset$ for any two distinct agents a_i and a_j (i.e., no resources are given to multiple agents). The set of all allocations for a MARA setting (A, R, U) is denoted by $\Pi_{A,R}$ and has cardinality n^m. We use the shorthand $u_i(X)$ to denote the utility $u_i(X(a_i))$ agent a_i can realize in allocation X because we assume agents not to be interested in externalities.

2.2 Representations of Utility Functions

Utility functions can be given in different ways, and the representation form potentially affects the complexity of the corresponding problems. We consider the following representation forms for utility functions:

1. **The bundle form:** A utility function $u : 2^R \to \mathbb{F}$ is in *bundle form* if it is represented by a list of pairs $(R', u(R'))$ for any bundle $R' \subseteq R$, omitting pairs with zero utility. This representation form is "fully expressive" (i.e., every utility function can be described in bundle form), but its drawback is a potentially exponential representation size in the number of resources.

2. **The k-additive form, for some fixed positive integer k:** A utility function $u : 2^R \to \mathbb{F}$ is in *k-additive form* if for each bundle $T \subseteq R$ with $\|T\| \leq k$, there is a unique coefficient $\alpha_T \in \mathbb{F}$ such that for every bundle $R' \subseteq R$ the following holds:

   $$u(R') = \sum_{T \subseteq R', \|T\| \leq k} \alpha_T.$$

   Sometimes we write (T, ℓ) for the coefficient $\alpha_T = \ell$. This coefficient expresses the "synergetic" value of some agent owning all the resources in T. This representation form is fully expressive only if k is large enough. On the other hand, choosing k to be relatively small allows for a relatively succinct representation of utility functions. Originally, Grabisch [11] defined the k-additive form. However, in multiagent resource allocation it was proposed for representing utilities by Chevaleyre et al. [6, 7] and, independently, in combinatorial auctions by Conitzer et al. [8].

3. **Straight-line program representation:** Informally, a straight-line program is a topologically sorted list of gates of a boolean circuit C that takes as input an m-dimensional binary vector and outputs s bits. Interpreting the input vector as a bundle of resources R' and the output as the binary representation of $u(R')$, we can say that C (or a corresponding straight-line program) represents utility function u.

   Formally (see, e.g., [9]), an (m, s)-*combinational logic network* is a directed graph with m input nodes $(\beta_1, \ldots, \beta_m)$ of in-degree 0, s output nodes $(\gamma_{s-1}, \ldots, \gamma_0)$ of out-degree 0, and gate nodes of in-degree at most 2 and out-degree at least 1. A gate node represents one of the common boolean operations $(\wedge, \vee, \neg)$. An input to the nodes $(\beta_1, \ldots, \beta_m)$ can be interpreted as a vector of length m and vice versa. Hence, every input vector β induces[3] an output vector

[3]Every bit at a gate node is induced as usual: If a is a gate node with a 2-ary boolean operation σ, then the bit induced at a is $b_1 \sigma b_2$, provided that (b_1, a) and (b_2, a) are edges of the graph, σ is a binary operation, and by b_1 and b_2 we mean the induced bits at nodes b_1 and b_2. For the boolean operation $\neg$, the definition is analogous.

$C(\beta)$, where we denote by $C(\beta)_i$ the i-th least significant bit of $C(\beta)$. Let $R = \{r_1, \ldots, r_m\}$, let $u : 2^R \to \mathbb{N}$ be a utility function and C an (m,s) combinational logic network. Denote by β_S the characteristic vector that has for every $j \in \{1, \ldots, m\}$ the j-th coordinate equal 1 if and only if $r_j \in S$ for some $S \subseteq R$. Utility function u is *realized* by C if for every $S \subseteq R$ with binary vector β_S the following holds:

$$u(S) = \sum_{i=0}^{s-1} 2^i \cdot C(\beta_S)_i.$$

The advantages of straight-line programs are mainly the efficiency of evaluation (linear time in the number of nodes) and its conciseness, which is supported by the following result by Pippenger and Fischer [22] and Schnorr [27].

Fact 1 *Let $f : \{0,1\}^m \to \{0,1\}^s$. If there exists a deterministic Turing machine that computes f in time T, then there exists a straight-line program of $\mathcal{O}(T \log T)$ lines that computes f as well.*

In multiagent resource allocation, utility representations by straight-line programs were introduced by Dunne et al. [9].

2.3 Measures of Social Welfare

The notion of social welfare is a tool to assess and rank allocations based on specific measures of quality. Thus, different allocations might be "the best allocation," depending on the notion of social welfare that is employed. We will study the following notions of social welfare.

Definition 2 *For a MARA setting (A,R,U) and an allocation $X \in \Pi_{A,R}$, define*

1. *the* utilitarian social welfare of X *as* $sw_u(X) = \sum_{a_i \in A} u_i(X)$;
2. *the* egalitarian social welfare of X *as* $sw_e(X) = \min_{a_i \in A}\{u_i(X)\}$;
3. *the* Nash product of X *as* $sw_N(X) = \prod_{a_i \in A} u_i(X)$.
4. *As an additional notation, for $S \in \{u,e,N\}$, denote the* maximum utilitarian/egalitarian/ Nash product social welfare of a MARA setting $M = (A,R,U)$ *(or of a problem instance that contains a MARA setting M) by*

$$\max_S(M) = \max\{sw_S(X) \mid X \in \Pi_{A,R}\}.$$

We write $\max(M)$ *for* $\max_S(M)$ *when S is clear from context.*

2.4 Problems Modeling Social Welfare Optimization

We are now ready to formally define the problems modeling social welfare optimization in multiagent resource allocation. We start with the decision problems. For $\mathbb{F} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}^+, \mathbb{Q}\}$ and form $\in \{\text{bundle}\} \cup \{k\text{-add} \mid k \geq 1\} \cup \{\text{SLP}\}$, where k-add abbreviates "k-additive" and SLP "straight-line program," define:

$\mathbb{F}$-NASH PRODUCT SOCIAL WELFARE OPTIMIZATION$_{\text{form}}$	
Given:	A MARA setting $M = (A,R,U)$, where form indicates how every $u_i : 2^R \to \mathbb{F}$ in U is represented, and a threshold $t \in \mathbb{F}$.
Question:	Is there an allocation $X \in \Pi_{A,R}$ such that $sw_N(X) \geq t$?

We abbreviate this problem by $\mathbb{F}$-NPSWO$_{\text{form}}$ (sometimes omitting the prefix "$\mathbb{F}$-"). The exact version of this problem is denoted by $\mathbb{F}$-EXACT NASH PRODUCT SOCIAL WELFARE OPTIMIZATION$_{\text{form}}$ (or, for short, by $\mathbb{F}$-XNPSWO$_{\text{form}}$) and asks, given a MARA setting $M = (A,R,U)$ and a target $t \in \mathbb{F}$, whether $\max_N(M) = t$.

The corresponding problems for utilitarian and egalitarian social welfare can be defined analogously and are abbreviated by $\mathbb{F}$-USWO$_{\text{form}}$ and $\mathbb{F}$-ESWO$_{\text{form}}$, respectively.

Apart from decision problems we also consider the corresponding three maximization problems, one for each type of social welfare. For example, the maximization problem for utilitarian social welfare is formally defined as follows:

$\mathbb{F}$-MAXIMUM UTILITARIAN SOCIAL WELFARE$_{\text{form}}$	
Input:	A MARA setting $M = (A,R,U)$, where form indicates how every $u_i : 2^R \to \mathbb{F}$ in U is represented.
Output:	$\max_u(M)$.

As a shorthand, write $\mathbb{F}$-MAX-USW$_{\text{form}}$. Based on sw_e and sw_N, $\mathbb{F}$-MAXIMUM EGALITARIAN SOCIAL WELFARE$_{\text{form}}$ (or $\mathbb{F}$-MAX-ESW$_{\text{form}}$) and $\mathbb{F}$-MAXIMUM NASH PRODUCT SOCIAL WELFARE$_{\text{form}}$ (or $\mathbb{F}$-MAX-NPSW$_{\text{form}}$) are defined accordingly.

2.5 Some Background on Complexity Theory and Approximation Theory

We assume basic knowledge of complexity theory, in particular of the complexity classes P, NP, and coNP, of central notions such as (polynomial-time many-one) reducibility (denoted by $\leq^{\text{p}}_{\text{m}}$), hardness and completeness of a problem for a complexity class with respect to $\leq^{\text{p}}_{\text{m}}$, etc. (see, e.g., the textbooks by Garey and Johnson [10], Papadimitriou [20], and Rothe [25]).

Papadimitriou and Yannakakis [21] introduced the complexity class DP, which consists of the differences of any two NP-problems. DP is the second level of the boolean hierarchy over NP and it is widely assumed that NP and coNP are both strictly contained in DP.

Typical DP problems are UNIQUE SATISFIABILITY ("*Does a given boolean formula have exactly one satisfying assignment?*") and exact variants of optimization problems such as the exact version of the TRAVELING SALESPERSON PROBLEM (EXACT-TSP): "*Given a graph and an integer t, does a shortest traveling salesperson tour have length exactly t?*" Intuitively, this problem is potentially harder than the usual TSP because both an NP problem ("*Does there exist a tour of length at most t, i.e., is the minimum tour length at most t?*," which *is* the usual TSP) and a coNP problem ("*Do all tours have length at least t, i.e., is the minimum tour length at least t?*") have to be solved to solve EXACT-TSP, which is complete for DP.

Turning to approximation theory, we define approximation algorithms for maximization problems and polynomial-time approximation schemes. Then we discuss reducibilities to prove inapproximability results.

Definition 3 (α-approximation algorithm) *Let Π be a maximization problem and $\alpha : \mathbb{N} \to (0,1)$. An* α-approximation algorithm *A for Π is a polynomial-time algorithm such that for each instance x of Π,*

$$val(A(x)) \geq \alpha(|x|) \cdot OPT(x),$$

where $val(A(x))$ denotes the value of a solution produced by A on input x and where $OPT(x)$ denotes the value of an optimal solution for x.

The *approximation factor* α might be a constant function such as $1-\varepsilon$ for some ε, $0 < \varepsilon < 1$, or a function of the input size, such as $1/\log n$ and $1/n^c$ for some $c > 0$.

Definition 4 (FPTAS) *A maximization problem* Π *has a* fully polynomial-time approximation scheme *(*FPTAS*) if for each* ε, $0 < \varepsilon < 1$, *there exists a* $(1-\varepsilon)$*-approximation algorithm* A_ε *for* Π, *where the running time is polynomial in* $1/\varepsilon$ *as well.*

One approach to prove inapproximability for a maximization problem is to find an α-gap-introducing reduction from an NP-complete problem.

Definition 5 (α-gap-introducing reduction) *Let* $A \subseteq \Sigma^*$ *be an* NP*-complete problem,* Π *be a maximization problem, and let* $\alpha : \mathbb{N} \to [0,1]$ *be a polynomial-time computable function of the input size. An* α-gap-introducing reduction from A to Π *is given by two polynomial-time computable functions* f *and* g *such that for each* $x \in \Sigma^*$,

1. $g(x)$ *is an instance of* Π,
2. *if* $x \in A$ *then* $OPT(g(x)) \geq f(x)$, *and*
3. *if* $x \notin A$ *then* $OPT(g(x)) < \alpha(|x|) \cdot f(x)$.

Note that an α-approximation algorithm B for a maximization problem Π that has an α-gap-introducing reduction from an NP-complete problem A implies $x \in A$ if and only if the value of the solution $B(g(x))$ is at least $\alpha(|x|) \cdot f(x)$. Hence, there can be no α-approximation algorithm for Π, unless P = NP.

Definition 6 (L-reduction) *Let* Π_1 *and* Π_2 *be some maximization problems. An* L-reduction from Π_1 to Π_2 *is given by two polynomial-time computable functions* f *and* g *and two parameters* α *and* β *such that for each instance* x *of* Π_1,

1. $y = f(x)$ *is an instance of* Π_2,
2. $OPT(y) \leq \alpha \cdot OPT(x)$, *and*
3. *for each solution* s_2 *for* y *of value* v_2, $s_1 = g(s_2)$ *is a solution for* x *of value* v_1 *such that*

$$OPT(x) - v_1 \leq \beta \cdot (OPT(y) - v_2).$$

Having an L-reduction from maximization problem Π_1 to Π_2 with parameters α,β and an $(1-\varepsilon)$-approximation algorithm for Π_2 implies a $(1-\alpha\beta\varepsilon)$-approximation algorithm for Π_1 by invoking f on the instance x of Π_1 to get an instance y of Π_2, then running the approximation algorithm for Π_2 on y and, at last, translating the solution back via g. Note that if Π_1 does not admit a $(1-\varepsilon)$-approximation algorithm and reduces to Π_2 with parameters $\alpha = \beta = 1$ then Π_2 cannot have a $(1-\varepsilon)$-approximation algorithm either.

For more background on approximation theory, see, e.g., the textbook by Vazirani [28] and the survey by Arora and Lund [1].

3 Related Work

The first paper concerned with classifying MARA problems in terms of their complexity is due to Chevaleyre et al. [6], see also [7]. They showed that the decision problem associated with utilitarian social welfare optimization is NP-complete for both the bundle and the k-additive form. Dunne et al. [9] proved that the problem remains NP-complete if utility functions are represented by straight-line programs. For further results on the complexity of fair allocation problems, we refer to Bouveret's thesis [3].

Roos and Rothe [24] proved NP-completeness for egalitarian social welfare optimization and social welfare optimization by the Nash product for the bundle form and for the k-additive form.

Table 1: Complexity of decision problems for (exact) social welfare optimization. Key: NP-c means "NP-complete" and DP-c means "DP-complete."

Social Welfare	Bundle	Reference	k-Additive	Reference
Utilitarian	NP-c	Chevaleyre et al. [7]	NP-c, $k \geq 2$	Chevaleyre et al. [7]
Egalitarian	NP-c	Roos & Rothe [24]	NP-c, $k \geq 1$	Roos & Rothe [24] and Lipton et al. [14]
Nash Product	NP-c	Roos & Rothe [24] and Ramezani & Endriss [23]	NP-c, $k \geq 1$	Roos & Rothe [24]
Exact Utilitarian Exact Egalitarian	DP-c	Roos & Rothe [24]	DP-c, $k \geq 2$	Roos & Rothe [24]
Exact Nash Product	DP-c	Theorem 7	DP-c, $k \geq 3$	Theorem 8

Social Welfare	SLP	Reference
Utilitarian	NP-c	Dunne et al. [9]
Egalitarian Nash Product	NP-c	Theorem 11

In addition, they proved DP-completeness for exact utilitarian and exact egalitarian social welfare optimization for both representation forms. Lipton et al. [14] provided a reduction to prove NP-hardness of finding a minimum-envy allocation (i.e., an allocation X that minimizes the envy $\max_{i,j}\{0, u_i(X(a_j)) - u_i(X(a_i))\}$). This reduction proves NP-hardness of the decision problem associated with egalitarian social welfare optimization as well. Independently of the result of Roos and Rothe [24], Ramezani and Endriss [23] proved the same NP-completeness result of Nash product social welfare optimization for the bundle form. Previous completeness results are summed up together with our results in Table 1.

Known approximability and inapproximability results in multiagent resource allocation have been surveyed recently in [19].

4 Complexity of Decision Problems Associated with Social Welfare Optimization

4.1 Utilities in the Bundle Form and the k-Additive Form

Roos and Rothe [24] conjectured that exact social welfare optimization by the Nash product is DP-complete for the bundle form and for the k-additive form. We confirm their conjecture in the affirmative.

It might be tempting to think that hardness for the decision problem associated with utilitarian social welfare optimization directly transfers to that for the Nash product by the straightforward reduction that maps utilities of value k to 2^k (cf. [23]). Note that not the exponential blow-up of the numbers encoding utilities causes a problem here, since the reduction from SET PACKING that Chevaleyre et al. [7] define to show NP-hardness of $\mathbb{Q}$-USWO$_{\text{bundle}}$ yield instances with utilities zero or one only. However, the reason for why this reduction doesn't work for the bundle form is that utilities of value zero that are omitted in the instances for utilitarian social welfare need to be encoded by the value $2^0 = 1$ in the resulting instance for the Nash product. In the worst case, this causes an exponential increase in the size of the instance constructed, and thus the reduction is not

polynomial-time.

Relatedly, another reason for why Nash product and utilitarian social welfare are not equivalent is that if we make the plausible assumption that the empty bundle has utility zero for everyone, the Nash product is trivially zero when there are fewer resources than agents (which implies that at least one agent must remain empty-handed), while utilitarian social welfare is not in that case.

Theorem 7 $\mathbb{Q}^+$-XNPSWO$_{\text{bundle}}$ *is* DP*-complete.*

Theorem 8 *For each* $k \geq 3$, $\mathbb{Q}^+$-XNPSWO$_{k\text{-add}}$ *is* DP*-complete.*

The proofs of Theorems 7 and 8 are omitted due to space limitations. In order to prove DP-hardness for $\mathbb{Q}^+$-XNPSWO$_{\text{bundle}}$, we need the following lemma by Chang and Kadin [4], who provided a sufficient condition for DP-hardness. It makes use of the definition of AND_2.

Definition 9 *Let* $L \subseteq \Sigma^*$ *be a decision problem.* L *has* AND_2 *if there exists a polynomial-time computable function* f *such that for all strings* $x, y \in \Sigma^*$*, it holds that*

$$x \in L \wedge y \in L \iff f(x,y) \in L.$$

Lemma 10 (Chang and Kadin [4]) *Let* $L \subseteq \Sigma^*$ *be a decision problem. If* L *is both* NP*-hard and* coNP*-hard and has* AND_2*, then* L *is* DP*-hard.*

We roughly present the idea of the proofs of Theorems 7 and 8. First, note that the proof of NP-hardness of $\mathbb{Q}^+$-XNPSWO$_{\text{bundle}}$ (see [24]) in fact proves coNP-hardness of this problem as well, and this reduction produces MARA settings, where the agents' utility functions take on binary values only. Since this is a special case of $\mathbb{Q}^+$-XNPSWO$_{\text{bundle}}$, hardness results carry over. To apply Lemma 10, it remains to show that any two instances can be merged in the sense of AND_2. Note that trivial merging of two $\mathbb{Q}^+$-XNPSWO$_{\text{bundle}}$ instances fails to prove AND_2: Consider instances M_1 and M_2 with target t_1 and t_2, respectively, with $t_1 < t_2$. Both instances are no-instances in that M_1 overachieves, i.e., $\max(M_1) > t_1$, and M_2 underachieves, i.e., $\max(M_2) < t_2$. However, the maximum of each instance equals the target of the other instance, that is, $\max(M_1) = t_2$ and $\max(M_2) = t_1$. If we trivially merged both instances, we would have a yes-instance with a greatest Nash product of $t_1 \cdot t_2$, the target of the merger. Therefore, we preprocess both input instances with a polynomial-time algorithm.

4.2 Utilities Represented by Straight-Line Programs

When utilities are represented by straight-line programs, we prove NP-completeness for egalitarian social welfare optimization and social welfare optimization by the Nash product. This helps to complete the picture for the complexity of social welfare optimization problems with straight-line program representation of utility functions, which Dunne et al. [9] initiated by their NP-completeness result for utilitarian social welfare optimization.

Theorem 11 $\mathbb{Q}$-ESWO$_{\text{SLP}}$ *and* $\mathbb{Q}^+$-NPSWO$_{\text{SLP}}$ *are* NP*-complete.*

Proof. Membership in NP is easy to see. To show NP-hardness, we reduce from the NP-complete problem MAX3SAT, which is formally defined as follows:

MAX3SAT	
Given:	A boolean formula φ in 3-CNF (i.e., in conjunctive normal form with three literals per clause) and $k \geq 2$.
Question:	Is there an assignment to the variables of φ such that at least k clauses are satisfied?

Let $\varphi = \bigwedge_{i=1}^{m}(z_i^1 \vee z_i^2 \vee z_i^3)$ be a given boolean formula in 3-CNF, where z_i^j, $1 \leq i \leq m$ and $j \in \{1,2,3\}$, is a literal of some variable $v \in V = \{v_1,\ldots,v_n\}$. Define $A = \{a_1,a_2\}$ and $R = \{r_1,\ldots,r_n,r_{n+1},\ldots,r_{2n}\}$. We say a bundle $S \subseteq R$ or its corresponding vector $\alpha_S = (x_1,\ldots,x_n,x_{n+1},\ldots,x_{2n})$ is *valid* if

$$\bigwedge_{i=1}^{n} XOR(x_i,x_{n+i}) = \bigwedge_{i=1}^{n}(\neg x_i \wedge x_{n+i}) \vee (x_i \wedge \neg x_{n+i}) = 1,$$

i.e., *XOR* denotes the boolean exclusive-or operation. Define a_1's utility function as

$$u_1(S) = \begin{cases} \text{number of satisfied clauses in } \varphi \text{ by } S & \text{if } S \text{ is valid} \\ 0 & \text{otherwise} \end{cases}$$

and a_2's utility function as

$$u_2 \equiv \begin{cases} m & \text{if we reduce to the egalitarian social welfare} \\ 1 & \text{if we reduce to social welfare by the Nash product.} \end{cases}$$

Write

$$u_1(\alpha_S) = \left(\bigwedge_{i=1}^{n} XOR(x_i,x_{n+i})\right) \cdot \sum_{i=1}^{m}(z_i^1 \vee z_i^2 \vee z_i^3),$$

where we replace[4] z_i^j by the corresponding value of x_k, $k \in \{1,\ldots,n\}$, if z_i^j is a positive literal of x_k; otherwise (that is, if z_i^j is a negated variable) we replace it by the value of x_{n+k}, $k \in \{1,\ldots,n\}$. By Proposition 1, we know there is an SLP of polynomial size that represents u_1.

Now consider a 3-CNF formula φ whose maximum number of satisfied clauses is k for some assignment $A : X \to \{0,1\}$. Assignment A induces an assignment vector $\alpha_S = (A(v_1),\ldots,A(v_n),1-A(v_1),\ldots,1-A(v_n))$. By definition, α_S is valid and a_1's utility is exactly k. The remaining resources go to a_2. Because a_2's utility can be ignored, the social welfare is a_1's utility, that is, the maximum number of satisfied clauses in φ.

For the other direction, note that we reduced from a legal 3-CNF formula. So there is an assignment that satisfies at least one clause. Hence, a_1 realizes a utility of at least one. Now let $k \geq 1$ be the maximum social welfare of this instance. By definition, $u_1(S) = k$ for some valid $\alpha_S = (x_1,\ldots,x_n,x_{n+1},\ldots,x_{2n})$. Truncating α_S by dropping the last n coordinates yields an assignment that satisfies k clauses. ❑

5 Approximability of Social Welfare Optimization

In the previous section, we have shown that the decision versions of certain social welfare optimization problems are intractable: either NP-complete for the standard problem that asks whether a given threshold of social welfare can be reached or exceeded in a given MARA setting, or DP-complete for the exact variant. It is natural to ask whether the optimization problems corresponding to these decision problems are intractable as well, or whether they allow efficient approximation schemes.

Known approximability and inapproximability results in multiagent resource allocation have been surveyed recently in [19]. Here we prove some novel results not included there. The first one is an inapproximability result about social welfare optimization by the Nash product for 1-additive utilities. We prove this result by a reduction from the well-known NP-complete problem EXACT COVER BY THREE SETS, which is defined as follows:

[4] Because we have a boolean circuit, we actually insert an edge (x_k,o_p^q), where o_p^q, $p \in \{1,\ldots,m\}$, $q \in \{1,2\}$, denotes the $\vee$-gate that is responsible for z_i^j in clause p.

EXACT COVER BY THREE SETS (X3C)	
Given:	A finite set B with $\|B\| = 3n$ and a collection $C = \{S_1, \ldots, S_m\}$ of 3-element subsets of B.
Question:	Does there exist a subcollection $C' \subseteq C$ such that every element of B occurs in exactly one of the sets in C'?

Theorem 12 *Assuming* P $\neq$ NP, MAX-NPSW$_{\text{1-add}}$ *cannot be approximated within a factor of* $2/3 + \varepsilon$ *for any* $\varepsilon > 0$.

Proof. Let (B,C) with $\|B\| = 3n$ and $C = \{S_1, \ldots, S_m\}$ be an instance of X3C. Without loss of generality, assume that $m \geq n$. Construct an instance $M = (A,R,U)$ of $\mathbb{Q}^+$-MAX-NPSW$_{\text{1-add}}$ as follows. Let A be a set of m agents, where agent a_i corresponds to S_i, and let $R = B \cup D$ be a set of $2n+m$ resources. That is, there are $3n$ "real" resources that correspond to the $3n$ elements of B, and there are $m-n$ "dummy" resources in D. Define the agents' utilities as follows. For each $a_i \in A$ and each $r_j \in R$, let

$$u_i(r_j) = \begin{cases} 1/3 & \text{if } r_j \in S_i \\ 1 & \text{if } r_j \in D \\ 0 & \text{otherwise.} \end{cases}$$

Also, define $u_i(\emptyset) = 0$ for all i, $1 \leq i \leq m$.

Suppose that (B,C) is a yes-instance of X3C. Then there exists a set $I \subseteq \{1, \ldots, m\}$, $\|I\| = n$, such that $S_i \cap S_j = \emptyset$ for all $i, j \in I$, $i \neq j$, and $\bigcup_{i \in I} S_i = B$. Hence, we assign the bundle S_i to agent a_i for each $i \in I$, and the dummy resources to the $m-n$ remaining agents. This allocation maximizes the Nash product social welfare, which now is at least 1. Furthermore, the sum of all agents' utilities is at most m. Hence, the product of the agents' individual utilities is maximal if and only if all agents have the same utility, which exactly equals 1.

Conversely, if (B,C) is a no-instance of X3C, we show that the maximum Nash product social welfare is at most $2/3$. Obviously, the sum of all agents' utilities is at most $m - 1/3$ in this case. The Nash product social welfare reaches the maximal value iff the utilities of the agents are as balanced as possible. The best allocation that satisfies this property is the following. Dummy resources are distributed to $m-n$ agents, $n-1$ agents get the $n-1$ disjoint bundles from $(S_1, \ldots, S_m)$, and the last agent is assigned the remaining bundle which has utility of at most $2/3$. This implies that $\max_N(M) \leq 2/3$. Therefore, an approximation algorithm with a factor better than $2/3$ will distinguish the "yes" and "no" instances of X3C. ❑

Theorem 12 shows that MAX-NPSW$_{\text{1-add}}$ cannot have a PTAS unless P = NP. This result also holds for MAX-ESW$_{\text{1-add}}$ due to Bezáková and Dani [2]. However, we show that there is an FPTAS for this problem whenever the number of agents is fixed, using a technique that was also used to give an FPTAS for a variety of scheduling problems (see [26] and [13]). We assume that for any agent a_i, the utility function u_i is nonnegative and $u_i(\emptyset) = 0$. The proof is omitted as well.

Theorem 13 *Both* MAX-NPSW$_{\text{1-add}}$ *and* MAX-ESW$_{\text{1-add}}$ *admit an FPTAS for any fixed number of agents.*

Coming back to the straight-line program representation of utility functions, notice that the reduction in the proof of Theorem 11 is an L-reduction with parameters $\alpha = \beta = 1$. There is a one-to-one correspondence between assignments of variables and assignments of resources to the first agent, where the maximum number of satisfied clauses equals the social welfare after the reduction. By setting the utility function of the second agent to the constant zero-function, we have a reduction with the same properties for the utilitarian case. Using the inapproximability result for Max3SAT by Håstad [12], we conclude:

Corollary 14 $\mathbb{Q}$-MAX-USW$_{\text{SLP}}$, $\mathbb{Q}$-MAX-ESW$_{\text{SLP}}$, $\mathbb{Q}^+$-MAX-NPSW$_{\text{SLP}}$ *are* NP*-hard to approximate within a factor of* $7/8 + \varepsilon$ *for every* $\varepsilon > 0$.

6 Conclusions

We have classified the decision versions of social welfare optimization problems for egalitarian and Nash product social welfare (for utilities represented by straight-line programs) and the exact variant for Nash product social welfare (for utilities in the bundle form and in the k-additive form) in terms of computational complexity. In addition, we have shown new approximability and inapproximability results for utilitarian, egalitarian, and Nash product social welfare. As interesting open problems for future work, we mention the study of complexity and approximability of social welfare optimization problems for different representation forms, improving approximation algorithms, and identifying tractable cases of restricted problem variants.

Acknowledgments: We gratefully acknowledge interesting discussions with Ulle Endriss, and we thank the COMSOC-2012, AAMAS-2012, ISAIM-2012, and STAIRS-2012 reviewers for their helpful comments.

References

[1] S. Arora and C. Lund. Hardness of approximations. In D. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, chapter 10, pages 399–446. PWS Publishing Company, 1996.

[2] I. Bezáková and V. Dani. Allocating indivisible goods. *SIGecom Exchanges*, 5(3):11–18, 2005.

[3] S. Bouveret. *Fair Allocation of Indivisible Items: Modeling, Computational Complexity and Algorithmics*. PhD thesis, Institut Supérieur de l'Aéronautique et de l'Espace, Toulouse, France, November 2007.

[4] R. Chang and J. Kadin. On computing boolean connectives of characteristic functions. *Theory of Computing Systems*, 28(3):173–198, 1995.

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

[6] Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet. Multiagent resource allocation with k-additive utility functions. In *Proceedings of the DIMACS-LAMSADE Workshop on Computer Science and Decision Theory*, volume 3 of *Annales du LAMSADE*, pages 83–100, 2004.

[7] Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet. Multiagent resource allocation in k-additive domains: Preference representation and complexity. *Annals of Operations Research*, 163:49–62, 2008.

[8] V. Conitzer, T. Sandholm, and P. Santi. Combinatorial auctions with k-wise dependent valuations. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 248–254. AAAI Press, 2005.

[9] P. Dunne, M. Wooldridge, and M. Laurence. The complexity of contract negotiation. *Artificial Intelligence*, 164(1–2):23–46, 2005.

[10] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[11] M. Grabisch. k-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, 92(2):167–189, 1997.

[12] J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

[13] E. Horowitz and S. Sahni. Exact and approximate algorithms for scheduling nonidentical processors. *Journal of the ACM*, 23(2):317–327, 1976.

[14] R. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 125–131. ACM Press, 2004.

[15] H. Moulin. *Fair Division and Collective Welfare*. MIT Press, 2004.

[16] N. Nguyen, T. Nguyen, M. Roos, and J. Rothe. Complexity and approximability of egalitarian and Nash product social welfare optimization in multiagent resource allocation. In *Proceedings of the 6th European Starting AI Researcher Symposium*. IOS Press, August 2012. To appear.

[17] N. Nguyen, T. Nguyen, M. Roos, and J. Rothe. Complexity and approximability of social welfare optimization in multiagent resource allocation (extended abstract). In *Proceedings of the 11th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1287–1288. IFAAMAS, June 2012.

[18] N. Nguyen, M. Roos, and J. Rothe. Exact optimization of social welfare by the Nash product is DP-complete. In *Website Proceedings of the 12th International Symposium on Artificial Intelligence and Mathematics*, January 2012.

[19] T. Nguyen, M. Roos, and J. Rothe. A survey of approximability and inapproximability results for social welfare optimization in multiagent resource allocation. In *Website Proceedings of the Special Session on Computational Social Choice at the 12th International Symposium on Artificial Intelligence and Mathematics*, January 2012.

[20] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, second edition, 1995.

[21] C. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28(2):244–259, 1984.

[22] N. Pippenger and M. Fischer. Relations among complexity measures. *Journal of the ACM*, 26(2):361–381, 1979.

[23] S. Ramezani and U. Endriss. Nash social welfare in multiagent resource allocation. In *Agent-Mediated Electronic Commerce. Designing Trading Strategies and Mechanisms for Electronic Markets*, pages 117–131. Springer-Verlag *Lecture Notes in Business Information Processing #79*, 2010.

[24] M. Roos and J. Rothe. Complexity of social welfare optimization in multiagent resource allocation. In *Proceedings of the 9th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 641–648. IFAAMAS, May 2010.

[25] J. Rothe. *Complexity Theory and Cryptology. An Introduction to Cryptocomplexity*. EATCS Texts in Theoretical Computer Science. Springer-Verlag, 2005.

[26] S. Sahni. Algorithms for scheduling independent tasks. *Journal of the ACM*, 23(1):116–127, 1976.

[27] C. Schnorr. The network complexity and the Turing machine complexity of finite functions. *Acta Informatica*, 7(1):95–107, 1976.

[28] V. Vazirani. *Approximation Algorithms*. Springer-Verlag, second edition, 2003.

Nhan-Tam Nguyen, Trung Thanh Nguyen, Magnus Roos, and Jörg Rothe
Institut für Informatik
Heinrich-Heine-Universität Düsseldorf
40225 Düsseldorf, Germany
Email: nhan-tam.nguyen@uni-duesseldorf.de
{thanh,roos,rothe}@cs.uni-duesseldorf.de