Exact Optimization of Social Welfare by the Nash Product is DP-Complete*

Nhan-Tam Nguyen
Institut für Informatik
Heinrich-Heine-Univ. Düsseldorf
40225 Düsseldorf, Germany

Magnus Roos
Institut für Informatik
Heinrich-Heine-Univ. Düsseldorf
40225 Düsseldorf, Germany

Jörg Rothe
Institut für Informatik
Heinrich-Heine-Univ. Düsseldorf
40225 Düsseldorf, Germany

Abstract

An important task in multiagent resource allocation, which provides mechanisms to allocate bundles of (indivisible and nonshareable) resources to agents, is to maximize social welfare. We study the computational complexity of exact social welfare optimization by the Nash product, which can be seen as a sensible compromise between the well-known notions of utilitarian and egalitarian social welfare. When utility functions are represented in the bundle or the k-additive form, for $k \geq 3$, we prove that the corresponding computational problems are DP-complete (where DP denotes the second level of the boolean hierarchy over NP), thus confirming two conjectures raised by Roos and Rothe (2010) in the affirmative.

Introduction

In multiagent resource allocation (see, e.g., the survey by Chevaleyre et al. (2006)), agents have preferences over bundles of resources. We consider preference representation by utility functions and assume that resources are indivisible and nonshareable. Taking the preferences of agents into account, the task is to allocate bundles of resources to agents.

By aggregating the agents' utilities we arrive at the notion of social welfare with which we can assess the quality of allocations from the viewpoint of a global system designer. One approach is the prominent *utilitarian social welfare*, which is the sum of the agents' utilities and which measures the average benefit every agent achieves. Utilitarian social welfare, however, lacks 'fairness' because the utilities that agents realize in a given allocation can differ greatly. Interpreting the utilities as bids or valuations in a combinatorial auction, utilitarian social welfare corresponds to an auctioneer's revenue.

Egalitarian social welfare, in contrast, looks at the agent that is worst off and seeks to improves this agent's utility. While this concept provides some measure of fairness when the minimum needs of all agents are to be satisfied, it does have some disadvantages; for example, it is not strictly monotonic: Raising the utility of an agent who is not worst off does not increase egalitarian social welfare.

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 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 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 (2004) presents further beneficial properties of the Nash product. For example, the Nash product is uniquely characterized by independence of individual scale of utilities,[1] i.e., even if different 'currencies' are used to measure the agents' utilities, the social welfare ordering remains unaffected.

Having a measure for the quality of allocations, it is a natural task to optimize social welfare, and to ask for the computational complexity of this task. Before doing so, the question has to be raised how utility functions are represented, as this might affect complexity as well. Reminiscent of the XOR bidding language in combinatorial auctions, the *bundle form* representation of a utility function is simply an enumeration of pairs of bundles and the utilities that are realized by a given bundle, omitting pairs with zero utility. This representation form is fully expressive, i.e., every utility function can be represented, but the representation can have exponential size. In contrast, the *k-additive form* of a utility function takes synergy effects into account and is a more succinct representation in many cases. This representation form has been introduced by Chevaleyre et al. (2008) (see also Chevaleyre et al. (2004)) in multiagent resource allocation and by Conitzer, Sandholm, and Santi (2005) in combinatorial auctions.

For both the bundle form and the k-additive form, NP-completeness is known for utilitarian social welfare optimization (Chevaleyre et al. 2008), egalitarian social welfare optimization form (Lipton et al. 2004; Roos and Rothe 2010) (see also Bansal and Sviridenko (2006)), and the Nash product (Roos and Rothe 2010; Ramezani and Endriss 2010). For the *exact* variants of these problems, DP-completeness is known for utilitarian and egalitarian social welfare optimization (Roos and Rothe 2010), where DP is the second level of the boolean hierarchy over NP.

We study the computational complexity of exact social

*This work was supported in part by DFG grant RO-1202/14-1.

[1] 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.

welfare optimization by the Nash product in the bundle form and the k-additive form, for $k \geq 3$. Confirming two conjectures raised by Roos and Rothe (2010) in the affirmative, we show that these problems are DP-complete as well.

Organization

In the Preliminaries section, we introduce the basic notions of multiagent resource allocation and complexity theory. Then we have a look at work that is related. Finally, we present our complexity results on exact social welfare optimization by the Nash product.

Preliminaries

Multiagent Resource Allocation

Slightly deviating from the notation of Chevaleyre et al. (2006), let $A = \{a_1, \ldots, a_n\}$ be the set of *agents*, $R = \{r_1, \ldots, r_m\}$ the set of *resources*, which each are assumed to be indivisible and nonshareable, and let $U = \{u_1, \ldots, u_n\}$ be the set of the agents' *utility functions*. The mapping $u_i : 2^R \to \mathbb{F}$ is agent a_i's utility function, 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 $\mathbb{Q}$ of rational numbers, and the set $\mathbb{Q}^+$ of nonnegative rational numbers). Such a triple (A, R, U) is called a *multiagent resource allocation setting* (a *MARA setting*, for short).

An *allocation* for a given MARA setting (A, R, U) is a mapping $X : A \to 2^R$ with $\bigcup_{a_i \in A} X(a_i) = R$ and $X(a_i) \cap X(a_j) = \emptyset$ for any two distinct agents a_i and a_j. 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.

We consider the following representation forms for utility functions:

1. The *bundle form*: A utility function u is represented by a list of pairs $(R', u(R'))$ for any bundle $R' \subseteq R$, where pairs with $u(R') = 0$ are dropped. This representation form is fully expressive, but the drawback is a potential exponential-size representation length.
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 there are coefficients[2] $\alpha_T \in \mathbb{F}$ for each bundle $T \subseteq R$ with $\|T\| \leq k$ such that for any bundle $R' \subseteq R$ the following holds:
$$u(R') = \sum_{T \subseteq R', \|T\| \leq k} \alpha_T$$
We write (T, ℓ) for the coefficient $\alpha_T = \ell$. For sufficiently large k, the k-additive form is fully expressive as well. The idea of this representation form is to express the additional value when receiving resources that fit together, or to express that agents might request a discount for large bundles. That is, agents might be willing to pay either more or less for a bundle of size up to k than the sum of their utilities for this bundle's single items.

[2] In addition to the standard assumptions in the literature, we require the coefficients to be elements of the codomain of the utility function. Note that computational hardness is inherited from more to less restricted problems, which thus leads to sharper results.

Definition 1 *Let (A, R, U) be a MARA setting and $X \in \Pi_{A,R}$ be an allocation.*

1. *The* utilitarian social welfare of X *is defined as*
$$sw_u(X) = \sum_{a_i \in A} u_i(X),$$
2. *the* egalitarian social welfare of X *is defined as*
$$sw_e(X) = \min\{u_i(X) \mid a_i \in A\},$$
and
3. *the* Nash product of X *is defined as*
$$sw_N(X) = \prod_{a_i \in A} u_i(X).$$

As an additional notation, denote the greatest Nash product of a MARA setting $M = (A, R, U)$ *(or of a problem instance that contains a MARA setting) by*
$$\max_N(M) = \max\{sw_N(X) \mid X \in \Pi_{A,R}\}.$$
We write $\max(M)$ *for* $\max_N(M)$ *when this is clear from context.*

For $\mathbb{F} \in \{\mathbb{N}, \mathbb{Q}^+, \mathbb{Q}\}$ and $\texttt{form} \in \{\texttt{bundle}, k\texttt{-additive}\}$, define the problem $\mathbb{F}$-NASH PRODUCT SOCIAL WELFARE OPTIMIZATION$_{\texttt{form}}$ (or $\mathbb{F}$-NPSWO$_{\texttt{form}}$, for short):

$\mathbb{F}$-NPSWO$_{\texttt{form}}$	
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 $t \in \mathbb{F}$.
Question:	Is there an allocation $X \in \Pi_{A,R}$ such that $sw_N(X) \geq t$ holds?

and the problem $\mathbb{F}$-EXACT NASH PRODUCT SOCIAL WELFARE OPTIMIZATION$_{\texttt{form}}$ (or $\mathbb{F}$-XNPSWO$_{\texttt{form}}$, for short):

$\mathbb{F}$-XNPSWO$_{\texttt{form}}$	
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 $t \in \mathbb{F}$.
Question:	Does it hold that $\max_N(M) = t$?

The corresponding problems for utilitarian and egalitarian social welfare can be defined analogously and have been studied, e.g., by Chevaleyre et al. (2006; 2008), and Roos and Rothe (2010).

Complexity Theory

We assume that the reader is familiar with the basic notions of computational complexity theory (see, e.g., the textbooks by Papadimitriou (1995) and Rothe (2005)), especially with the complexity classes P (deterministic polynomial-time), NP (nondeterministic polynomial-time), coNP (containing all problems whose complement is in NP), the polynomial-time many-one reducibility (denoted by $\leq^{\mathrm{p}}_{\mathrm{m}}$), and the notions of hardness and completeness for a complexity class with respect to $\leq^{\mathrm{p}}_{\mathrm{m}}$.

Papadimitriou and Yannakakis (1984) introduced the complexity class $\mathrm{DP} = \{L_1 - L_2 \,|\, L_1, L_2 \in \mathrm{NP}\}$, which contains 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?*"[3]) 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. Another problem known to be DP-complete is EXACT-4-COLORABILITY (Rothe 2003); see, e.g., Cai and Meyer (1987), Wagner (1987), Riege and Rothe (2006) and Rothe (2005) for further DP-completeness results.

Related Work and Our Results

Table 1(a) shows the known and new complexity results for social welfare optimization and exact social welfare optimization in the bundle form, and Table 1(b) shows them for the *k*-additive form. In particular, Chevaleyre et al. (2008) proved that utilitarian social welfare optimization is NP-complete for both the bundle form and the *k*-additive form. Dunne, Wooldridge, and Laurence (2005) proved the same result for another representation form, the so-called "SLP form," where utility functions are represented by "straight-line programs." Confirming a conjecture of Chevaleyre et al. (2006), Roos and Rothe (2010) proved that egalitarian social welfare optimization is NP-complete in both the bundle and the *k*-additive form,[4] and they showed the same for social welfare optimization by the Nash product. Ramezani and Endriss (2010) proved NP-completeness of Nash product social welfare optimization in the bundle form as well, independently of Roos and Rothe (2010).

In addition, confirming another conjecture of Chevaleyre et al. (2006), Roos and Rothe (2010) showed that both exact utilitarian and exact egalitarian social welfare optimization in the bundle form and the *k*-additive form are DP-complete. The complexity of *exact* social welfare optimization by the Nash product was left open by Roos and Rothe (2010), who conjectured that this problem is DP-complete in both the bundle form and the *k*-additive form as well. We solve this conjecture in the affirmative. Note that it is not immediately clear whether the exact version of an NP-complete optimization problem is DP-complete; e.g., EXACT-3-COLORABILITY is NP-complete (Rothe 2003).

Table 1: Complexity of (exact) social welfare optimization. Key: NP-c means "NP-complete" and DP-c means "DP-complete." Results marked by [†] are due to Chevaleyre et al. (2008), results marked by [⋆] are due to Roos and Rothe (2010), where the result marked by [‡] has been shown independently by Ramezani and Endriss (2010) and the result marked by [¶] is shown by a reduction that Lipton et al. (2004) used for a different problem (see also Bansal and Sviridenko (2006)).

(a) Bundle form

	Social Welfare	Exact Social Welfare
Utilitarian	NP-c [†]	DP-c [⋆]
Egalitarian	NP-c [⋆]	DP-c [⋆]
Nash Product	NP-c [‡, ⋆]	DP-c [Thm. 8]

(b) *k*-additive form

	Social Welfare	Exact Social Welfare
Utilitarian	NP-c, $k \geq 2$ [†]	DP-c, $k \geq 2$ [⋆]
Egalitarian	NP-c, $k \geq 1$ [¶, ⋆]	DP-c, $k \geq 2$ [⋆]
Nash Product	NP-c, $k \geq 1$ [⋆]	DP-c, $k \geq 3$ [Thm. 11]

Hardness results do not carry over from utilitarian social welfare optimization to social welfare optimization by the Nash product by using the fact that $\prod u_i = \sum \log u_i$ because such a reduction from the utilitarian problem to the Nash product problem takes exponential time, if the utilities are unbounded. However, reducing from social welfare optimization by the Nash product to utilitarian social welfare optimization by taking logarithms yields an upper bound for the Nash product problem. But if the utilities are constants, a reduction from the utilitarian case is possible to prove hardness lower bounds, which is credited to an anonymous reviewer and Endriss (personal communication, November 2011). Our proofs of Theorems 8 and 11 present an alternative approach that is independent of hardness results for utilitarian social welfare optimization. Furthermore, by using Chang and Kadin's technique to show DP-hardness, our proofs can potentially be extended to similar problems that are believed to be hard for $\mathrm{P}_{||}^{\mathrm{NP}}$, the class of problems solvable in polynomial time with parallel (i.e., truth-table) access to an NP oracle,[5] by showing that they additionally have 'infinite' *OR* (see Chang and Kadin (1995)).

[3] This problem is known to be coNP-hard and in DP but not known to be DP-complete—equivalently, as noted by Blass and Gurevich (1982), not known to be NP-hard.

[4] Egalitarian social welfare optimization (in the *k*-additive form) is known as the "Santa Claus problem" in the field of combinatorial auctions. Bansal and Sviridenko (2006) studied the approximability of this optimization problem, referring to Lipton et al. (2004) for NP-hardness of the corresponding decision problem. The (simple) reduction from PARTITION that Lipton et al. (2004) use to show that the problem of finding a minimum-envy allocation is NP-hard can indeed also be used to show NP-hardness of egalitarian social welfare optimization in the *k*-additive form.

[5] This complexity class was introduced by Papadimitriou and Zachos (1983). For more specific details and results regarding this class, we refer to the survey by Hemaspaandra, Hemaspaandra, and Rothe (1997) and the references cited therein.

Complexity of Exact Social Welfare Optimization by the Nash Product

In order to show our DP-completeness results, we need the following lemma by Chang and Kadin (1995), who provided a sufficient condition for DP-hardness. It makes use of the definition of AND_2: A problem has AND_2 if any two strings (encoding two instances of this problem) can be merged in polynomial time such that the merger is a yes-instance if and only if both input instances are yes-instances.

Definition 2 *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 3 (Chang and Kadin (1995)) *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 prove DP-completeness for $\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$.[6] As mentioned above, Roos and Rothe (2010) proved that $\mathbb{Q}^+$-NPSWO$_{\texttt{bundle}}$ is NP-complete. In fact, their proof shows NP-hardness and coNP-hardness for the exact variant $\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$ 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$_{\texttt{bundle}}$, hardness results carry over.

To apply Lemma 3, it remains to show that any two instances can be merged in the sense of AND_2. To this end, we preprocess both input instances with the following polynomial-time algorithm, which will be used both for the bundle and the 3-additive form. Intuitively, the algorithm adds a new Nash product to the set of all possible Nash products by adding new agents and resources. When adding the 'right' Nash product, we shift the greatest Nash product, which is needed for merging two no-instances because such a merger may become a yes-instance, violating the equivalence required by AND_2.

Let $\texttt{form} \in \{\texttt{bundle}, \texttt{3-additive}\}$. The algorithm takes as input an $\mathbb{N}$-XNPSWO$_{\texttt{form}}$ instance (A,R,U,t), where $A = \{a_1,\ldots,a_n\}$, $R = \{r_1,\ldots,r_m\}$, $U = \{u_1,\ldots,u_n\}$, and a rational $\varepsilon \in (0,1)$. For each agent $a_i \in A$, generate an '*empty set simulator*' $r_{\|R\|+i} \in R_\emptyset$. In addition, create *identity resources* $\mathrm{id}_i \in R_{\mathrm{id}}$ for each original resource $r_i \in R$ and each 'empty set simulator.' Create two resources, $\mathrm{id}_0 \in R_{\mathrm{id}}$ and $r_0 \in R$. For all original resources, identity resources, and 'empty set simulators,' generate *signaling resources* $\mathrm{sig}_r \in R_{\mathrm{sig}}$. For each agent $a_i \in A$, set the utility of the bundle containing only the 'empty set simulator' $r_{\|R\|+i}$ equal to the utility of the bundle containing no resources. Set the utility of this bundle over the empty set to zero. Furthermore, the utility of the bundle with the corresponding identity resource id_i is set to one. Add a new control agent, $\overline{a_c}$, with utility one for the bundle $\{\mathrm{id}_0, \mathrm{sig}_{r_0}\}$ and with utility one for the bundle $\{r_0, \mathrm{sig}_{\mathrm{id}_0}\}$. For each original resource and 'empty set simulator' r_i, add a new agent $\overline{a_i}$ with utility one for the bundle $\{\mathrm{sig}_{r_{i-1}}, \mathrm{id}_i, \mathrm{sig}_{\mathrm{id}_i}\}$ and with utility one for the bundle $\{\mathrm{sig}_{\mathrm{id}_{i-1}}, r_i, \mathrm{sig}_{r_i}\}$. When utilities are represented in the bundle form, add additional *garbage collector agents* $\overline{g}_i$ and resources $\mathrm{id}_{g_i} \in R_{\mathrm{id}}$ for each 'empty set simulator,' identity resource, and for $\mathrm{sig}_{r_{\|R\|+\|A\|}}$ and $\mathrm{sig}_{\mathrm{id}_{\|R\|+\|A\|}}$. Agent $\overline{g}_i$'s utilities are one for the bundle $\{\mathrm{id}_{g_i}\}$ and for the bundle that contains the agent's identity resource and the corresponding 'empty set simulator,' identity or signaling resource. At last, add a new agent $\overline{a}$. Depending on the target t of the input, the utilities of agent $\overline{a}$ differ:

- If the target t is positive, set the utility of the bundle $\{\mathrm{id}_0\}$ to one and set the utility of the bundle $\{r_0\}$ to $t-\varepsilon$. The target t^p of the output is set equal to t.
- If the target t is zero, set the utility of the bundle $\{\mathrm{id}_0\}$ to two and set the utility of the bundle $\{r_0\}$ to one. The target t^p of the output is set to 1.

Output the modified instance (A^p, R^p, U^p, t^p) with agents

$$\begin{aligned} A^p &= A \cup \{\overline{a}, \overline{a_c}\} \cup \{\overline{a_i} \mid 1 \le i \le \|R\| + \|A\|\} \cup \\ &\quad \{\overline{g_i} \mid 1 \le i \le 2\|A\| + \|R\| + 2\} \end{aligned}$$

and resources $R^p = R \cup R_\emptyset \cup R_{\mathrm{id}} \cup R_{\mathrm{sig}}$, and with utilities in U^p and target vector t^p as described above.

Example 4 *Our input to the preprocessing algorithm is the given $\mathbb{N}$-XNPSWO$_{\texttt{3-additive}}$ instance $M_1 = (A,R,U,22)$ with $A = \{a_1,a_2,a_3\}$, $R = \{r_1,\ldots,r_4\}$, and the set U of utility functions as shown in Table 2, and an $\varepsilon \in (0,1)$.*

After preprocessing, we obtain the output $(A^p, R^p, U^p, 22)$ with agents $A^p = A \cup \{\overline{a}, \overline{a_c}\} \cup \{\overline{a_i} \mid 1 \le i \le \|R\| + \|A\|\}$ and resources

$$\begin{aligned} R^p &= R \cup \{r_{\|R\|+i} \mid 1 \le i \le \|A\|\} \cup \\ &\quad \{id_i, sig_{r_i}, sig_{id_i} \mid 0 \le i \le \|R\| + \|A\|\} \cup \{r_0\}, \end{aligned}$$

and the utility functions in U^p as shown in Table 3.

Table 2: Input to the preprocessing algorithm in Example 4

agent	utilities
a_1	$(\{r_1\},1)$, $(\{r_1,r_4\},5)$, $(\emptyset,1)$
a_2	$(\{r_2\},1)$, $(\{r_3\},2)$, $(\{r_2,r_3,r_4\},3)$
a_3	$(\{r_4\},4)$

The following lemma proves the correctness of the preprocessing algorithm, which adds new Nash products to the Nash products that were reachable before preprocessing. If the target t is positive, new Nash products of zero and of $t-\varepsilon$ for some chosen $\varepsilon \in (0,1)$ become reachable. If the target t is zero, new reachable Nash products are zero and one, in addition to all Nash products of the input instance (before preprocessing) multiplied by a constant factor.

Lemma 5 *Let (A,R,U,t) be a given $\mathbb{N}$-XNPSWO$_{\texttt{form}}$ instance, where $\texttt{form} \in \{\texttt{bundle}, \texttt{3-additive}\}$, and let $\varepsilon \in (0,1)$. Let (A^p, R^p, U^p, t^p) be the corresponding preprocessed instance. Define $Y = \{sw_N(X) \mid X \in \Pi_{A,R}\}$ and $Y' = \{sw_N(X) \mid X \in \Pi_{A^p,R^p}\}$.*

1. *For $t > 0$, $Y' = Y \cup \{0, t-\varepsilon\}$.*

[6] Note that negative utilities are not sensible in the context of the Nash product.

Table 3: Output of the preprocessing algorithm in Example 4

agent	utilities
a_1	$(\{r_1\},1)$, $(\{r_1,r_4\},5)$, $(\{r_5\},1)$, $(\{\mathrm{id}_1\},1)$
a_2	$(\{r_2\},1)$, $(\{r_3\},2)$, $(\{r_2,r_3,r_4\},3)$, $(\{r_6\},0)$, $(\{\mathrm{id}_2\},1)$
a_3	$(\{r_4\},4)$, $(\{r_7\},0)$, $(\{\mathrm{id}_3\},1)$
$\overline{a}$	$(\{r_0\},22-\varepsilon)$, $(\{\mathrm{id}_0\},1)$
$\overline{a_c}$	$(\{\mathrm{sig}_{r_0},\mathrm{id}_0\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_0},r_0\},1)$
$\overline{a_1}$	$(\{\mathrm{sig}_{r_0},\mathrm{id}_1,\mathrm{sig}_{\mathrm{id}_1}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_0},r_1,\mathrm{sig}_{r_1}\},1)$
$\overline{a_2}$	$(\{\mathrm{sig}_{r_1},\mathrm{id}_2,\mathrm{sig}_{\mathrm{id}_2}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_1},r_2,\mathrm{sig}_{r_2}\},1)$
$\overline{a_3}$	$(\{\mathrm{sig}_{r_2},\mathrm{id}_3,\mathrm{sig}_{\mathrm{id}_3}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_2},r_3,\mathrm{sig}_{r_3}\},1)$
$\overline{a_4}$	$(\{\mathrm{sig}_{r_3},\mathrm{id}_4,\mathrm{sig}_{\mathrm{id}_4}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_3},r_4,\mathrm{sig}_{r_4}\},1)$
$\overline{a_5}$	$(\{\mathrm{sig}_{r_4},\mathrm{id}_5,\mathrm{sig}_{\mathrm{id}_5}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_4},r_5,\mathrm{sig}_{r_5}\},1)$
$\overline{a_6}$	$(\{\mathrm{sig}_{r_5},\mathrm{id}_6,\mathrm{sig}_{\mathrm{id}_6}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_5},r_6,\mathrm{sig}_{r_6}\},1)$
$\overline{a_7}$	$(\{\mathrm{sig}_{r_6},\mathrm{id}_7,\mathrm{sig}_{\mathrm{id}_7}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_6},r_7,\mathrm{sig}_{r_7}\},1)$

2. *For $t=0$, $Y' = \{2sw_N(X) \mid X \in \Pi_{A,R}\} \cup \{0,1\}$.*

PROOF. Let $X \in \Pi_{A,R}$ be a legal allocation. The extension of X to $X' : A^p \to 2^{R^p}$ is defined as follows:

- If $X(a_i) = \emptyset$, then $X'(a_i) = \{r_{\|R\|+i}\}$.
- If $X(a_i) = R'$ and $R' \neq \emptyset$, then $X'(a_i) = R'$.
- Set $X'(\overline{a}) = \{\mathrm{id}_0\}$, $X'(\overline{a_c}) = \{r_0, \mathrm{sig}_{\mathrm{id}_0}\}$, and $X'(\overline{a_i}) = \{\mathrm{sig}_{r_{i-1}}, \mathrm{id}_i, \mathrm{sig}_{\mathrm{id}_i}\}$ for $1 \leq i \leq \|R\| + \|A\|$.
- For the bundle form, assign all remaining resources to the garbage collector agents.

It is easy to see that X' is a legal allocation. Each of the agents $\overline{a_c}$ and $\overline{a_i}$, $1 \leq i \leq \|R\| + \|A\|$, and each garbage collector agent, if utilities are represented in the bundle form, realizes a utility of 1. Agent $\overline{a}$ realizes a utility of 1 if $t > 0$, and a utility of 2 if $t = 0$. If an agent receives the empty set, this assignment is substituted by the corresponding 'empty set simulator.' Since all resources have to be allocated and new resources have been introduced, garbage collector agents take excess resources ('empty set simulators'; the remaining identity resources if $\overline{a}$ received r_0; $\mathrm{sig}_{r_{\|R\|+\|A\|}}$ or $\mathrm{sig}_{\mathrm{id}_{\|R\|+\|A\|}}$, depending on the allocation to $\overline{a}$) when utilities are represented in the bundle form. For the k-additive form, garbage collector agents are not needed because the coefficients of utility functions are nonnegative and hence the utility functions are monotonic. Thus, the Nash product of X' is $sw_N(X') = sw_N(X)$ if $t > 0$, and it is $sw_N(X') = 2sw_N(X)$ if $t = 0$. Since new resources have been added, there are new allocations with a Nash product of 0. The allocation X'' with

- $X''(a_i) = \{\mathrm{id}_i\}$ for all $a_i \in A$,
- $X''(\overline{a}) = \{r_0\}$,
- $X''(\overline{a_c}) = \{\mathrm{id}_0, \mathrm{sig}_{r_0}\}$, and
- $X''(\overline{a_i}) = \{\mathrm{sig}_{\mathrm{id}_{i-1}}, r_i, \mathrm{sig}_{r_i}\}$ for all $\overline{a_i} \in A^p$

(and assigning all remaining resources to the garbage collector agents for the bundle form) has a Nash product of $sw_N(X'') = t - \varepsilon$ if $t > 0$, and a Nash product of $sw_N(X'') = 1$ if $t = 0$. We thus have $Y \cup \{0, t-\varepsilon\} \subseteq Y'$ if $t > 0$, and $\{2sw_N(X) \mid X \in \Pi_{A,R}\} \cup \{0,1\} \subseteq Y'$ if $t = 0$.

For the converse, look at agent $\overline{a}$. Either $\{\mathrm{id}_0\}$ or $\{r_0\}$ can be assigned to this agent. When assigning both of these resources to $\overline{a}$, agent $\overline{a_c}$ realizes a utility of 0, no matter whether we are working with the bundle form or the 3-additive form representation. If agent $\overline{a}$ receives $\{\mathrm{id}_0\}$, then agent $\overline{a_c}$ has to get $\{r_0, \mathrm{sig}_{\mathrm{id}_0}\}$. Consequently, in order to realize a positive utility, agent $\overline{a_1}$ has to be assigned $\{\mathrm{sig}_{r_0}, \mathrm{id}_1, \mathrm{sig}_{\mathrm{id}_1}\}$. Hence, agent $\overline{a_2}$ gets $\{\mathrm{sig}_{r_1}, \mathrm{id}_2, \mathrm{sig}_{\mathrm{id}_2}\}$, and so on. This way agent $a_i \in A$ cannot receive $\{\mathrm{id}_i\}$. Therefore the restriction of such an allocation X to A is an element of $\Pi_{A,R}$ with a social welfare of $sw_N(X)$ if $t > 0$, and with half this social welfare if $t = 0$ (because agent $\overline{a}$'s utility is neglected in this case). Else if agent $\overline{a}$ receives $\{r_0\}$, agent $\overline{a_c}$ gets $\{\mathrm{id}_0, \mathrm{sig}_{r_0}\}$. Then all agents $\overline{a_i}$, $1 \leq i \leq \|R\| + \|A\|$, take away all resources and 'empty set simulators,' leaving the identity resources for the original agents $a_i \in A$ and the garbage collectors agents. Overall, such an allocation has a Nash product of $t - \varepsilon$ if $t > 0$, and a Nash product of 1 if $t = 0$, establishing the lemma for the bundle form.

For the 3-additive form, we change the proof slightly. We extend an allocation $X \in \Pi_{A,R}$ to X' similarly, but if an agent receives a bundle different from the empty set, the agent's 'empty set simulator' is added to the bundle received because the empty set is a subset of every set. ❑

Corollary 6 follows immediately from Lemma 5. It shows how the greatest Nash product changes after adding new allocations by preprocessing.

Corollary 6 *Let $I = (A,R,U,t)$ be an $\mathbb{N}$-XNPSWO$_{\texttt{form}}$ instance, where $\texttt{form} \in \{\texttt{bundle}, \texttt{3-additive}\}$, let $\varepsilon \in (0,1)$, and let $J = (A^p, R^p, U^p, t^p)$ be its preprocessed instance.*

1. *For $t > 0$,*
   (a) *if $\max(I) = t$ then $\max(J) = t$,*
   (b) *if $\max(I) > t$ then $\max(J) > t$,*
   (c) *if $\max(I) < t$ then $\max(J) = t - \varepsilon < t$.*
2. *For $t = 0$,*
   (a) *if $\max(I) = t$ then $\max(J) = t^p = 1$,*
   (b) *if $\max(I) > t$ then $\max(J) > t^p = 1$.*

Before we show how to merge instances as required, we introduce an additional notation. Let $M_1 \oplus M_2$ denote the merger of two $\mathbb{F}$-XNPSWO$_{\texttt{form}}$ instances, $M_1 = (A_1, R_1, U_1, t_1)$ and $M_2 = (A_2, R_2, U_2, t_2)$, by taking the componentwise disjoint union of A_1 and A_2, of R_1 and R_2, and of U_1 and U_2, and set $t_{M_1 \oplus M_2} = t_1 \cdot t_2$. That is, if the sets of agents or resources are not disjoint, we make them disjoint by renaming agents or resources. Note that the agents' utilities for bundles R' that contain new resources are zero for the bundle form because no additional pairs $(R', u(R'))$ are specified. For the k-additive form, the utilities of such bundles R' is equal to the utility of the intersection of the old resources and R'. Hence, the merger has a greatest Nash product of

$$\max(M_1 \oplus M_2) = \max(M_1)\max(M_2)$$

if all utilities are nonnegative.

Lemma 7 $\mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$ *has AND*$_2$*, where* $\texttt{form} \in \{\texttt{bundle}, \texttt{3-additive}\}$*. That is, there exists a polynomial-time computable function* f *satisfying that for any two* $\mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$ *instances* $M_1 = (A_1, R_1, U_1, t_1)$ *and* $M_2 = (A_2, R_2, U_2, t_2)$*, it holds that*

$$M_1 \in \mathbb{Q}^+\text{-XNPSWO}_{\texttt{form}} \wedge M_2 \in \mathbb{Q}^+\text{-XNPSWO}_{\texttt{form}} \iff f(M_1, M_2) \in \mathbb{Q}^+\text{-XNPSWO}_{\texttt{form}}.$$

PROOF. Without loss of generality, we may assume to have $\mathbb{N}$-XNPSWO$_{\texttt{form}}$ instances $M_1 = (A_1, R_1, U_1, t_1)$ and $M_2 = (A_2, R_2, U_2, t_2)$, because we can convert every given $\mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$ instance into a corresponding $\mathbb{N}$-XNPSWO$_{\texttt{form}}$ instance by appropriate multiplication in polynomial time. Apply the preprocessing algorithm to both instances separately with an $\varepsilon \in (0,1)$ that is to be determined later on, and use Corollary 6 for all cases.

First suppose that $t_1 \cdot t_2 > 0$. After preprocessing, we have $M_i^p = (A_i^p, R_i^p, U_i^p, t_i)$, $i \in \{1,2\}$. Merge M_1^p and M_2^p to obtain $M_1^p \oplus M_2^p$, and consider the following cases:

1. $M_1^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. Then we have $\max(M_1^p \oplus M_2^p) = \max(M_1^p) \cdot \max(M_2^p) = t_1 \cdot t_2$, as required.
2. $M_1^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$.
   (a) If $\max(M_2^p) > t_2$ then $\max(M_1^p \oplus M_2^p) = t_1 \cdot \max(M_2^p) > t_1 \cdot t_2$.
   (b) If $\max(M_2^p) < t_2$ then $\max(M_1^p \oplus M_2^p) = t_1(t_2 - \varepsilon) < t_1 \cdot t_2$ for each ε, $0 < \varepsilon < 1$.
3. $M_1^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. This case can be handled analogously to the above case.
4. $M_1^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$.
   (a) If $\max(M_1^p) > t_1$ and $\max(M_2^p) > t_2$ then we immediately have $\max(M_1^p \oplus M_2^p) > t_1 \cdot t_2$.
   (b) If $\max(M_1^p) < t_1$ and $\max(M_2^p) < t_2$ then we immediately have $\max(M_1^p \oplus M_2^p) = (t_1 - \varepsilon)(t_2 - \varepsilon) < t_1 \cdot t_2$ for each ε, $0 < \varepsilon < 1$.
   (c) If $\max(M_1^p) > t_1$ and $\max(M_2^p) < t_2$ then define $\delta = \max(M_1^p) - t_1$. Note that $\delta \in [1, \infty) \subseteq \mathbb{N}$. The maximum social welfare by the Nash product is $\max(M_1^p \oplus M_2^p) = (t_1 + \delta)(t_2 - \varepsilon) = t_1 \cdot t_2 - t_1 \cdot \varepsilon + t_2 \cdot \delta - \varepsilon \cdot \delta$. For which ε does this equal $t_1 \cdot t_2$? Exactly if $-t_1 \cdot \varepsilon + t_2 \cdot \delta - \varepsilon \cdot \delta = 0$, which in turn holds if and only if $\delta = {t_1 \cdot \varepsilon}/{(t_2 - \varepsilon)}$. As $\delta \geq 1$, the equality does not hold if ${(t_1 \cdot \varepsilon)}/{(t_2 - \varepsilon)} < 1$, i.e., exactly if $\varepsilon < {t_2}/{(t_1 + 1)}$. Using such an ε, we can ensure $\max(M_1^p \oplus M_2^p) \neq t_1 \cdot t_2$, as desired.
   (d) If $\max(M_1^p) < t_1$ and $\max(M_2^p) > t_2$ then, arguing as above, the criterion to choose ε is $\varepsilon < {t_1}/{(t_2 + 1)}$.

Summing up, for this case ε has to be chosen so that $0 < \varepsilon < \min\{{t_2}/{(t_1+1)}, {t_1}/{(t_2+1)}\}$.

Next suppose that $t_1 > 0$ and $t_2 = 0$. After preprocessing, we have $M_1^p = (A_1^p, R_1^p, U_1^p, t_1)$ and $M_2^p = (A_2^p, R_2^p, U_2^p, t_2^p)$. Merging them gives $M_1^p \oplus M_2^p$. Consider the following cases:

1. $M_1^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. Then we have $\max(M_1^p \oplus M_2^p) = t_1 \cdot t_2^p = t_1$, as required.
2. $M_1^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. If $\max(M_2^p) > t_2^p$ (note that $\max(M_2^p) < t_2^p$ never occurs), then $\max(M_1^p \oplus M_2^p) = t_1 \cdot \max(M_2^p) > t_1 \cdot t_2^p = t_1$, since $\max(M_2^p) > 1$.
3. $M_1^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$.
   (a) If $\max(M_1^p) > t_1$ then $\max(M_1^p \oplus M_2^p) = \max(M_1^p) \cdot 1 > t_1 \cdot t_2^p = t_1$.
   (b) If $\max(M_1^p) < t_1$ then $\max(M_1^p \oplus M_2^p) = (t_1 - \varepsilon) \cdot 1 < t_1 \cdot t_2^p = t_1$ for each ε, $0 < \varepsilon < 1$.
4. $M_1^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$.
   (a) If $\max(M_1^p) > t_1$ and $\max(M_2^p) > t_2^p$ then we immediately have $\max(M_1^p \oplus M_2^p) > t_1 \cdot t_2^p = t_1$.
   (b) If $\max(M_1^p) < t_1$ and $\max(M_2^p) > t_2^p$ then we immediately have $\max(M_1^p \oplus M_2^p) = (t_1 - \varepsilon)\max(M_2^p)$. This is equal to t_1 exactly if $(t_1 - \varepsilon)2\max(M_2) = t_1$, which in turn is true if and only if $\max(M_2) = {t_1}/{2(t_1 - \varepsilon)}$. Since $\max(M_2) \geq 1$, the equality does not hold if ${t_1}/{2(t_1 - \varepsilon)} < 1$, i.e., exactly if $\varepsilon < {t_1}/{2}$. As $t_1 \geq 1$, using an ε with $\varepsilon < 1/2$, $\max(M_1^p \oplus M_2^p) \neq t_1$ follows.

Hence, for this case ε has to be chosen so that $0 < \varepsilon < 1/2$.

The case "$t_1 = 0$ and $t_2 > 0$" with $M_1^p = (A_1^p, R_1^p, U_1^p, t_1^p)$ and $M_2^p = (A_2^p, R_2^p, U_2^p, t_2)$ can be handled analogously to the above case.

Finally, suppose that both $t_1 = 0$ and $t_2 = 0$. After preprocessing, we now have $M_i^p = (A_i^p, R_i^p, U_i^p, t_i^p)$, $i \in \{1,2\}$. Merge them to obtain $M_1^p \oplus M_2^p$, and consider the following cases:

1. $M_1^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. Then we have $\max(M_1^p \oplus M_2^p) = t_1^p \cdot t_2^p = 1$, as required.
2. $M_1^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. If $\max(M_2^p) > t_2^p$ (note that $\max(M_2^p) < t_2^p$ never occurs) then $\max(M_1^p \oplus M_2^p) = 1 \cdot \max(M_2^p) > t_1^p \cdot t_2^p = 1$ because $\max(M_2^p) > 1$.
3. $M_1^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \in \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. This case can be handled analogously to the above case.
4. $M_1^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}} \wedge M_2^p \notin \mathbb{Q}^+$-XNPSWO$_{\texttt{form}}$. If $\max(M_1^p) > t_1^p$ and $\max(M_2^p) > t_2^p$, then $\max(M_1^p \oplus M_2^p) > t_1^p \cdot t_2^p = 1$, since $\max(M_1^p) > 1$ and $\max(M_2^p) > 1$.

This completes the proof. ❑

Theorem 8 $\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$ *and* $\mathbb{Q}$-XNPSWO$_{\texttt{bundle}}$ *are* DP*-complete.*

PROOF. Note that upper bounds are inherited downward with respect to $\leq_{\mathrm{m}}^{\mathrm{p}}$, and lower bounds are inherited upward with respect to $\leq_{\mathrm{m}}^{\mathrm{p}}$. Thus, as $\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$ is a special case of $\mathbb{Q}$-XNPSWO$_{\texttt{bundle}}$ (and therefore $\leq_{\mathrm{m}}^{\mathrm{p}}$-reduces to it), it is enough to prove a DP upper bound for $\mathbb{Q}$-XNPSWO$_{\texttt{bundle}}$ and a DP-hardness lower bound for $\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$.

$\mathbb{Q}$-XNPSWO$_{\texttt{bundle}}$ is in DP because it can be written as

$$\mathbb{Q}\text{-XNPSWO}_{\texttt{bundle}} = A \cap B,$$

Table 4: Input to the preprocessing algorithm in Example 9

agent	utilities
a_1^*	$(\{r_1^*\},2)$, $(\{r_1^*,r_3^*\},2)$
a_2^*	$(\{r_1^*,r_2^*,r_3^*\},3)$, $(\{r_2^*,r_3^*\},2)$

Table 5: Output of the preprocessing algorithm in Example 9

agent	utilities
a_1^*	$(\{r_1^*\},2)$, $(\{r_1^*,r_3^*\},2)$, $(\{r_4^*\},0)$, $(\{\mathrm{id}_1^*\},1)$
a_2^*	$(\{r_1^*,r_2^*,r_3^*\},3)$, $(\{r_2^*,r_3^*\},2)$, $(\{r_5^*\},0)$, $(\{\mathrm{id}_2^*\},1)$
$\overline{a^*}$	$(\{r_0^*\},6-\varepsilon)$, $(\{\mathrm{id}_0^*\},1)$
$\overline{a_c^*}$	$(\{\mathrm{sig}_{r_0^*},\mathrm{id}_0^*\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_0^*},r_0^*\},1)$
$\overline{a_1^*}$	$(\{\mathrm{sig}_{r_0^*},\mathrm{id}_1^*,\mathrm{sig}_{\mathrm{id}_1^*}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_0^*},r_1^*,\mathrm{sig}_{r_1^*}\},1)$
$\overline{a_2^*}$	$(\{\mathrm{sig}_{r_1^*},\mathrm{id}_2^*,\mathrm{sig}_{\mathrm{id}_2^*}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_1^*},r_2^*,\mathrm{sig}_{r_2^*}\},1)$
$\overline{a_3^*}$	$(\{\mathrm{sig}_{r_2^*},\mathrm{id}_3^*,\mathrm{sig}_{\mathrm{id}_3^*}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_2^*},r_3^*,\mathrm{sig}_{r_3^*}\},1)$
$\overline{a_4^*}$	$(\{\mathrm{sig}_{r_3^*},\mathrm{id}_4^*,\mathrm{sig}_{\mathrm{id}_4^*}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_3^*},r_4^*,\mathrm{sig}_{r_4^*}\},1)$
$\overline{a_5^*}$	$(\{\mathrm{sig}_{r_4^*},\mathrm{id}_5^*,\mathrm{sig}_{\mathrm{id}_5^*}\},1)$, $(\{\mathrm{sig}_{\mathrm{id}_4^*},r_5^*,\mathrm{sig}_{r_5^*}\},1)$

where

$$A=\left\{(A,R,U,t)\;\middle|\;\begin{array}{l}(A,R,U)\text{ is a MARA setting, }t\in\mathbb{Q},\\ \text{and }\max\{sw_N(X)\,|\,X\in\Pi_{A,R}\}\geq t\end{array}\right\}$$

is in NP, whereas

$$B=\left\{(A,R,U,t)\;\middle|\;\begin{array}{l}(A,R,U)\text{ is a MARA setting, }t\in\mathbb{Q},\\ \text{and }\max\{sw_N(X)\,|\,X\in\Pi_{A,R}\}\leq t\end{array}\right\}$$

is in coNP.

$\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$ is DP-hard by using Lemma 3 and Lemma 7 together with the NP-hardness and coNP-hardness results for $\mathbb{Q}^+$-XNPSWO$_{\texttt{bundle}}$ of Roos and Rothe (2010) that were mentioned right below Lemma 3. ❑

Example 9 *In addition to the instance from Example 4 consider the following* $\mathbb{N}$-XNPSWO$_{\texttt{3-additive}}$ *instance* $M_2=(A,R,U,6)$ *with* $A=\{a_1^*,a_2^*\}$, $R=\{r_1^*,r_2^*,r_3^*\}$, *and the set* U *of utility functions as shown in Table 4, and an* $\varepsilon\in(0,1)$. *Note that the agents and resources of* M_2 *have already been renamed to ensure that we have disjunct agent and resource sets when merging. It holds that* $\max(M_1)=24$ *and* $\max(M_2)=4$. *(Recall that utilities are given in* 3*-additive form. For example,* $\max(M_1)=24$ *is realized by assigning resource* r_1 *and also the empty bundle to agent* a_1, *resources* r_2 *and* r_3 *to agent* a_2, *and resource* r_4 *to agent* a_3, *which gives a Nash product of* $(1+1)(1+2)4=24$.*)*

After merging the preprocessed instances M_1^p *and* M_2^p *with the new target* $t=22\cdot 6=132$, *the greatest Nash product is* $\max(M_1^p\oplus M_2^p)=24(6-\varepsilon)$. *Although both* M_1^p *and* M_2^p *(or* M_1 *and* M_2*) are no-instances, choosing a 'wrong'* ε *such as* $\varepsilon=1/2$ *gives* $\max(M_1^p\oplus M_2^p)=132$. *But if* ε *is chosen such that* $0<\varepsilon<\min\{22/(6+1),6/(22+1)\}$ *holds, e.g.,* $\varepsilon=1/4$, *we have* $\max(M_1^p\oplus M_2^p)=138>132$.

To show DP-hardness of $\mathbb{Q}^+$-XNPSWO$_{\texttt{3-additive}}$, note that the NP-hardness proof for $\mathbb{Q}^+$-NPSWO$_{\texttt{1-additive}}$ by Roos and Rothe (2010) works also for the exact variant, $\mathbb{Q}^+$-XNPSWO$_{\texttt{1-additive}}$, and this proof can be modified to show coNP-hardness as well. They reduce from the NP-complete problem PARTITION (see, e.g., Garey and Johnson (1979)). We prove coNP-hardness by reducing from the complement of a restricted variant of PARTITION:

PARTITION$_{\geq 2}$	
Given:	A sequence $(c_1,\ldots,c_n)$ of integers with $c_i\geq 2$, $1\leq i\leq n$, such that $\sum_i c_i$ is even.
Question:	Does there exist a subset $I\subseteq\{1,\ldots,n\}$ such that $\sum_{i\in I}c_i=\sum_{i\notin I}c_i$?

It is easy to see that PARTITION$_{\geq 2}$ is NP-complete as well.

Lemma 10 $\mathbb{N}$-XNPSWO$_{\texttt{3-additive}}$ *is* coNP*-hard.*

PROOF. We reduce from the complement of PARTITION$_{\geq 2}$. Let $(c_1,\ldots,c_n)$ be a given PARTITION$_{\geq 2}$ instance with $\sum c_i=C$. Similarly to the proof of Roos and Rothe (2010), create agents a_1 and a_2. In addition, create agents $\{\overline{a},\overline{a_c}\}\cup\{\overline{a_i}\,|\,1\leq i\leq n\}$ and resources $R=\{r_i,\mathrm{id}_i,\mathrm{sig}_{\mathrm{id}_i},\mathrm{sig}_{r_i}\,|\,0\leq i\leq n\}$. Set $u_k(\{r_i\})=c_i$ and $u_k(\{\mathrm{id}_k\})=1$ for $k\in\{1,2\}$ and $1\leq i\leq n$. For agent $\overline{a}$ set $u_{\overline{a}}(\{r_0\})=(C/2)^2-1$ and $u_{\overline{a}}(\{\mathrm{id}_0\})=1$, and for agent $\overline{a_c}$ set $u_{\overline{a_c}}(\{\mathrm{sig}_{r_0},\mathrm{id}_0\})=1$ and $u_{\overline{a_c}}(\{\mathrm{sig}_{\mathrm{id}_0},r_0\})=1$. Finally, agents $\overline{a_i}$, $1\leq i\leq n$, have utilities $u_{\overline{a_i}}(\{\mathrm{sig}_{\mathrm{id}_{i-1}},r_i,\mathrm{sig}_{r_i}\})=1$ and $u_{\overline{a_i}}(\{\mathrm{sig}_{r_{i-1}},\mathrm{id}_i,\mathrm{sig}_{\mathrm{id}_i}\})=1$. Set $t=(C/2)^2-1$.

To show the equivalence, suppose that $(c_1,\ldots,c_n)\in$ PARTITION$_{\geq 2}$. Then there is an allocation under which agent a_1 and agent a_2 realize $C/2$ utility each. To do that, agent $\overline{a}$ needs to get $\{\mathrm{id}_0\}$, and the remaining $\overline{a_i}$ get bundles containing the identity resources only. Hence, the greatest Nash product is $(C/2)^2\neq t$.

If $(c_1,\ldots,c_n)\notin$ PARTITION$_{\geq 2}$: For each allocation $X\in\Pi_{A,R}$ that assigns $\{\mathrm{id}_0\}$ to $\overline{a}$, either a_1 or a_2 realizes a utility of $C/2+d_X$ and the other agent realizes a utility of $C/2-d_X$ for some d_X. Because it is a PARTITION$_{\geq 2}$ instance, we have $d_X\geq\min_i c_i\geq 2$. So every allocation that assigns $\{\mathrm{id}_0\}$ to $\overline{a}$ has a Nash product of at most $(C/2)^2-2^2$. Since it is a maximizing allocation procedure, agent $\overline{a}$ receives $\{r_0\}$ instead. Arguing similarly to the proof of Lemma 5, the greatest Nash product is $(C/2)^2-1=t$. ❑

Theorem 11 *For each* $k\geq 3$, $\mathbb{Q}^+$-XNPSWO$_{k\texttt{-additive}}$ *and* $\mathbb{Q}$-XNPSWO$_{k\texttt{-additive}}$ *are* DP*-complete.*

PROOF. The proof of membership in DP is similar to the proof of Theorem 8. DP-hardness follows from Lemma 3, the NP-hardness result of Roos and Rothe (2010) that shows that $\mathbb{Q}^+$-XNPSWO$_{\texttt{1-additive}}$ is NP-hard as well, and from Lemmas 7 and 10. Since 3-additive utilities are k-additive utilities for $k>3$ (Conitzer, Sandholm, and Santi 2005), $\mathbb{Q}$-XNPSWO$_{k\texttt{-additive}}$ for $k\geq 3$ is DP-complete. ❑

Remark 12 *By multiplying all numbers by their least common multiple, we can slightly improve upon the hardness result to show that the problems* $\mathbb{N}$-XNPSWO$_{\texttt{bundle}}$ *and* $\mathbb{N}$-XNPSWO$_{k\text{-}\texttt{additive}}$ *for* $k \geq 3$ *are* DP*-complete as well.*

Conclusions

We have shown that exact social welfare optmization by the Nash product is DP-complete for two representations of the agents' utility functions, namely the bundle and the k-additive form, for $k \geq 3$. Interesting questions that remain open concern the computational complexity of social welfare optimization (for the different notions of social welfare) when utility functions are represented by "straight-line programs" (Chevaleyre et al. 2006; Dunne, Wooldridge, and Laurence 2005).

Particularly interesting are approxamibility and inapproximability results for social welfare optimization problems, and we propose this topic for future research. Previous results in this line of research have been surveyed by Nguyen, Roos, and Rothe (2012).

Acknowledgments

We gratefully acknowledge interesting discussions with Ulle Endriss, and we thank the anonymous reviewers for their helpful comments.

References

Bansal, N., and Sviridenko, M. 2006. The Santa Claus problem. In *Proceedings of the 38th ACM Symposium on Theory of Computing*, 31–40. ACM Press.

Blass, A., and Gurevich, Y. 1982. On the unique satisfiability problem. *Information and Control* 55(1–3):80–88.

Cai, J., and Meyer, G. 1987. Graph minimal uncolorability is D^P-complete. *SIAM Journal on Computing* 16(2):259–277.

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

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

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

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

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

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

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

Hemaspaandra, E.; Hemaspaandra, L.; and Rothe, J. 1997. Raising NP lower bounds to parallel NP lower bounds. *SIGACT News* 28(2):2–13.

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

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

Nguyen, T.; Roos, M.; and Rothe, J. 2012. 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*.

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

Papadimitriou, C., and Zachos, S. 1983. Two remarks on the power of counting. In *Proceedings of the 6th GI Conference on Theoretical Computer Science*, 269–276. Springer-Verlag *Lecture Notes in Computer Science #145*.

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

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

Riege, T., and Rothe, J. 2006. Completeness in the boolean hierarchy: Exact-Four-Colorability, minimal graph uncolorability, and exact domatic number problems – a survey. *Journal of Universal Computer Science* 12(5):551–578.

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

Rothe, J. 2003. Exact complexity of Exact-Four-Colorability. *Information Processing Letters* 87(1):7–12.

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

Wagner, K. 1987. More complicated questions about maxima and minima, and some closures of NP. *Theoretical Computer Science* 51:53–80.