Nash Social Welfare in Multiagent Resource Allocation

Sara Ramezani and Ulle Endriss

Institute for Logic Language and Computation (ILLC)
University of Amsterdam

Abstract. We study different aspects of the multiagent resource allocation problem when the objective is to find an allocation that maximizes Nash social welfare, the product of the utilities of the individual agents. The Nash solution is an important welfare criterion that combines efficiency and fairness considerations. We show that the problem of finding an optimal outcome is NP-hard for a number of different languages for representing agent preferences; we establish new results regarding convergence to Nash-optimal outcomes in a distributed negotiation framework; and we design and test algorithms similar to those applied in combinatorial auctions for computing such an outcome directly.

1 Introduction

Multiagent resource allocation (MARA) is a loosely defined research area concerned with the study of mechanisms for distributing a set of resources among a group of agents—typically software agents with limited reasoning capabilities [3]. Each agent has their own preferences (e.g., a utility function) over the bundles of items they may receive, and the perceived quality of a chosen outcome (allocation of items to agents) will depend on these individual preferences.

One quality indicator is (economic) *efficiency*. The most basic criterion is Pareto efficiency: an outcome is Pareto efficient if there is no other outcome that would make one agent better off without harming any of the others. A stronger criterion is based on utilitarian social welfare: an allocation is optimal in this sense if it maximizes the sum of the utilities of the individual agents. Both criteria have been widely used in Artificial Intelligence, Multiagent Systems and Electronic Commerce. Some recent work in these disciplines has also recognized the fact that the desideratum of finding efficient allocations needs to be balanced with appropriate *fairness* considerations [2, 8, 12], a dilemma that has long been discussed in Economics, Political Science, and Philosophy [13]. Fairness criteria include egalitarian social welfare (measuring quality in terms of the utility experienced by the poorest agent) and envy-freeness (an allocation is envy-free if no agent would want to change bundle with any of the others).

In this paper we focus on a criterion that combines aspects of efficiency and fairness, the *Nash social welfare* criterion [13]. An outcome maximizes Nash social welfare if it maximizes the product of the individual agent utilities. This idea goes back to John Nash's famous solution to the bargaining problem [14]. While the Nash solution is recognized as being of central importance in the Economics literature at large, in this paper we shall study it in the context of two approaches to MARA in which it has received little or no attention to date.

The first of these is a *distributed approach* where allocations emerge as the agents negotiate a sequence of (typically small) local deals, one deal at a time. This approach has been studied in detail with utilitarian social welfare as the criterion of choice [19, 8, 7], and to a lesser extent also for various fairness criteria [8, 4]. Mirroring known results for other welfare criteria, we show how to define a local criterion for assessing the acceptability of a deal that will guarantee that a Nash-optimal allocation emerges eventually, and we highlight some of the limitations of the approach by proving results on the structural complexity of deals and the length of deal sequences required.

The second approach is based on considering the MARA problem as a problem faced by some central authority rather than the agents themselves. This approach is inspired by *combinatorial auctions* [6]. The standard winner determination problem in combinatorial auctions is equivalent to the problem of finding an allocation with maximal utilitarian social welfare, if bids are taken to reflect actual agent utility. By slightly changing the objective function in the winner determination problem we arrive at the problem of computing a Nash-optimal allocation. We exploit this correspondence and important techniques from the combinatorial auction literature to tackle the problem.

The remainder of this paper is organized as follows. Section 2 collects basic definitions on MARA, social welfare, and preference representation. Section 3 then argues why Nash social welfare is such an important criterion for measuring the quality of resource allocations and gives complexity results that show that computing a Nash-optimal allocation is NP-hard. Section 4 presents our results on convergence and the length of deal sequences in the distributed MARA framework, while Section 5 presents our winner determination algorithms for "Nash combinatorial auctions" and provides some experimental results. Section 6 concludes. (Due to space limitations, some proofs are only sketched and two have been omitted entirely, but full proofs are available elsewhere [17].)

2 Preliminaries

In this section we introduce the MARA framework we shall use, and review basic definitions regarding social welfare and preference representation languages.

2.1 Multiagent Resource Allocation

A *multiagent resource allocation scenario* is defined as a triple $\langle \mathcal{A}, \mathcal{R}, \mathcal{U} \rangle$. $\mathcal{A}$ is a finite set of n agents $\mathcal{A} = \{1, \ldots, n\}$; $\mathcal{R}$ is a finite set of m resources $\mathcal{R} = \{r_1, \ldots, r_m\}$; and $\mathcal{U} = \{u_1, \ldots, u_n\}$ is a set of utility functions, one for each agent. Each $u_i \in \mathcal{U}$ is a mapping from sets of resources to the nonnegative reals: $u_i : 2^{\mathcal{R}} \rightarrow \mathbb{R}^+ \cup \{0\}$. The utility functions model the preferences of the agents over alternative bundles. An *allocation* $A : \mathcal{A} \to 2^{\mathcal{R}}$ is a function mapping agents to sets of resources, such that $A(i) \cap A(j) = \emptyset$ if $i \neq j$, and $\bigcup_{i=1}^{n} A(i) = \mathcal{R}$. $A(i)$ represents the resources that agent i owns in allocation A, the resources are non-sharable. That is, each allocation A is a partitioning of all of the resources among the agents. The utility agent i assigns to allocation A is $u_i(A) = u_i(A(i))$, by a slight abuse of notation since agents only care about the resources that they obtain (not others' bundles).

2.2 Collective Utility Functions and Social Welfare Orderings

We now review several criteria for assessing social welfare [13]. A *utility profile* for an allocation A is a vector that contains the utility of all agents, i.e., a vector $u(A) = (u_1(A), \cdots, u_n(A)) \in \mathbb{R}^n$. A *collective utility function* (CUF) is a function $W : \mathbb{R}^n \to \mathbb{R}$ mapping utility profiles to the reals. Most social welfare criteria can be defined as a CUF. For each CUF, we obtain a corresponding *social welfare ordering* (SWO), a transitive and complete binary preference relation on utility profiles, by defining the ordering such that one profile is preferred over another if and only if it has higher collective utility. As each allocation induces a utility profile, we apply the notions of CUF and SWO also to allocations.

The *utilitarian CUF* is defined as the sum of the utilities of all agents; in our framework: $sw_u(A) = \sum_{i \in \mathcal{A}} u_i(A)$. Hence, the utilitarian SWO favors allocations in which agents have higher average utility.

The *egalitarian CUF*, on the other hand, defines the lowest utility of the profile as the collective utility; here: $sw_e(A) = \min\{u_i(A) \mid i \in \mathcal{A}\}$. Hence, the egalitarian SWO prefers profiles in which the worst-off agent is better off.

A drawback of the utilitarian CUF is that it only takes average utility levels into account, and has no reservations for fairness or equality. It would, for instance, prefer a utility profile $(100, 1)$ to $(50, 50)$, although the latter is clearly much closer to equality with a very small cost for the total utility. Conversely, the egalitarian CUF is insensitive to change in overall welfare, as long as the worst-off agent has better circumstances, e.g. it prefers $(25, 25)$ to $(24, 76)$ although the second doubles the utility level with a very small cost for the worst-off agent.

The *Nash CUF* is defined as the product of the individual agent utilities: $sw_N(A) = \prod_{i \in \mathcal{A}} u_i(A)$. It balances efficiency and fairness, e.g. it prefers $(50, 50)$ to $(100, 1)$ and $(24, 76)$ to $(25, 25)$, and so it has neither of the pitfalls of the two previously mentioned CUFs. More formal properties of the Nash CUF will be discussed in Section 3.

2.3 Preference representation languages

So far we have only defined an agent's utility abstractly, as a function u mapping bundles to numerical values. Any concrete implementation will have to use a *preference representation language* for encoding u [3]. In the context of combinatorial auctions, for example, this is the role of the bidding language [15]. We now briefly review the languages for which we shall give algorithms in Section 5 and complexity results in Section 3.2.

The most basic language is the *explicit form*; it simply lists for each bundle S the value $u(S)$, unless that value is 0. The *XOR-language* from the combinatorial auction literature [20] is essentially the same language, except that here an implicit monotonicity assumption is being made: a bid such as $\langle S_1, p_1\rangle$ XOR $\cdots$ XOR $\langle S_n, p_n\rangle$, where the $S_i \in 2^{\mathcal{R}}$ are bundles and the $p_i \in \mathbb{R}^+$ are prices, defines the utility function u with $u(X) = \max_{1\leq i\leq n}\{p_i \mid X \supseteq S_i\}$. The *OR-language* is similar: here the value of a bundle is the maximal sum of prices achievable by selecting a set of non-overlapping atomic bids [15].

The *language of positive cubes* belongs to the family of languages based on weighted propositional formulas [22]. The basic idea is to identify resources with propositional variables. A *goalbase* G is a set of pairs $\{(\varphi_j, w_j)\}_j$, where each φ_j is a propositional formula over these variables and each $w_j \in \mathbb{R}$ is a weight. For the language of positive cubes, each φ_j is required to be a conjunction of positive literals. Then the utility function u induced by goalbase G is defined via $u(X) = \sum\{w \mid (\varphi, w) \in G \text{ and } X \models \varphi)\}$.[1] For example, given the goalbase $\{(p, 5), (p \wedge q, 2)\}$, our agent would value bundle $\{p\}$ at 5, bundle $\{q\}$ at 0, and bundle $\{p, q\}$ at 7. Positive cubes of length $\leq k$ are equivalent to the well-known class of k-additive functions [10]. We will be specifically interested in the restriction of the language where all weights have to be positive.

3 Nash Social Welfare: Axiomatics and Complexity

In this section we discuss some properties of the Nash CUF/SWO defined earlier. We review its axiomatic characterization and we analyze its complexity.

3.1 Axiomatic Characterization of Nash Social Welfare

The Nash SWO encourages both increases in overall utility in a society and improved equality among agents, which is an important factor for assessing fairness and collective welfare. More formally, the Nash SWO can be characterized by a set of axioms that represent some generally desirable properties for an SWO, and thus justify its position as one of the most important social welfare orderings in the literature.

Before we review these axioms, note that it is only meaningful to define the Nash SWO when the utility functions are nonnegative, since if some utilities are negative, the outcome of the Nash CUF would not be continuous (with respect to changes in the individual utilities), and would fluctuate unreasonably between positive and negative values depending on whether the number of individuals with negative utilities is even or odd. Also, for some of our results, it is required for the utility profiles to be strictly positive, because a single zero in the profile would make the rest of the profile irrelevant, which is not desirable. Sometimes replacing zeros with very small positive values can be useful, as it keeps the value of the CUF low, without losing sensitivity to the other values in the profile.

[1] The consequence relation $\models$ is defined in the obvious way; for positive cubes φ it amounts to $X \models \varphi$ if and only if each of the conjuncts in φ is an element of X.

Following Moulin [13], we now sketch the unique properties of the Nash SWO. The Nash SWO, along with the utilitarian SWO and a generalized version of the egalitarian SWO (called the *leximin ordering*) are the main representatives of a family of social welfare orderings characterized by three important axioms:

- *Independence of unconcerned agents*: Agents whose utility levels are the same in two utility profiles should not have an effect on the ordering of the profiles. This is in line with the assumption that agents' happiness depends on their own utilities only.
- *Pigou-Dalton transfer principle*: An SWO that satisfies this principle prefers or is at least indifferent to any change that involves only two agents and that is both mean-preserving and inequality-reducing as far as the utilities of these two agents are concerned. This is the most basic fairness principle.
- *Independence of common utility scale*: Re-scaling the utility functions of all agents with the same factor should not affect the SWO. That is, the ordering should not change if *all* agents scale their utilities with the same factor, e.g., if they use a different currency.

In addition to these axioms, which are clearly reasonable and desirable for an SWO, the Nash SWO is uniquely identified by being *independent of the individual scale of utilities*: the SWO remains the same even if each agent rescales his utility function using a *different* factor, This property ensures that the Nash CUF eliminates the possibility of manipulation by means of changing the scale used by an individual agent; examples for this can be found in [13, 17].

3.2 Computational complexity

Next we analyse the computational complexity of the problem of finding an allocation that maximises Nash social welfare. We shall assume basic familiarity with the theory of NP-completeness [16]. The decision variant of the Nash Welfare Optimisation problem is defined as follows:

Nash Welfare Optimisation (NASH)	
Given:	Resource allocation scenario $\langle \mathcal{A}, \mathcal{R}, \mathcal{U} \rangle$ and $K \in \mathbb{Q}$.
Question:	Is there an allocation A such that $sw_N(A) > K$?

The precise complexity of NASH will depend on the language used to represent $\mathcal{U}$. First, consider the related problem MAXUTIL: given a *single* utility function u and some $K \in \mathbb{Q}$, is there a bundle S such that $u(S) > K$? MAXUTIL is known to be NP-hard for a number of representation languages, including the OR-language [18] and the language of positive cubes with arbitrary (positive and negative) weights [22]. For these languages, we obtain NP-hardness of NASH as an immediate corollary (because MAXUTIL is equivalent to NASH for societies consisting of just a single agent). On the other hand, for the XOR-language, for the explicit form, and for positive cubes with positive weights, MAXUTIL is trivial and cannot be used to establish complexity results for NASH.

We now prove complexity results for the two languages for which we shall provide algorithms in Section 5.

Theorem 1. NASH *is NP-complete if the XOR-language is used, even when each agent only assigns a value to a single bundle.*

Proof. If each agent only values a single bundle, then the XOR-language is equivalent to the OR-language; and for this case Utilitarian Welfare Optimisation is known to be NP-hard [18, 3]. Given that there is a polynomial reduction from that problem to NASH (replace each price p in the bids by 2^p), NP-hardness of NASH follows. NP-membership also holds: if an oracle produces an allocation A, it is possible to verify in polynomial time whether $sw_N(A) > K$. □

Theorem 2. NASH *is NP-complete if the language of positive cubes with positive weights is used, even when each agent only assigns a value to a single bundle.*

Proof. Again, for the case where each agent only attaches a non-zero weight to a single bundle (cube), positive cubes with positive weights work exactly as the OR-language. Hence, the same proof as given for Theorem 1 applies. □

While the proofs given are basically the same, we emphasise that neither of the two theorems entails the other. This is so, because neither of the two languages is as succinct as the other in all cases [5].

4 Distributed Approach

We now apply the tools of a distributed approach to MARA [19, 8, 7] to the problem of optimizing Nash social welfare. In this framework, there is no central authority and agents are free to negotiate allocations by means of deals over some of their exchanging resources. The main appeal of such a model is that the agents can be designated as independent entities (e.g., software programs) that make decisions based on their locally available information only. The computational costs can also be distributed among the agents in this way. Throughout this section, it is assumed that all utility functions are strictly positive.

4.1 Deals

We assume that the agents start out in an *initial allocation*, and can agree on deals that result in changes in the allocation. A *deal* $\delta = (A, A')$ is a an ordered pair of distinct allocations (before/after). The set of agents *involved in* δ is $\mathcal{A}^\delta = \{i \in \mathcal{A} | A(i) \neq A'(i)\}$. This definition of a deal is very broad and may involve the reallocation many resources among many agents.

4.2 Convergence and Necessity

Previous work has shown that deals that satisfy *myopic individual rationality* are sufficient for guaranteeing that any sequence of such deals will result in an allocation that maximizes the utilitarian CUF [19]. Here we try to obtain a similar result for the Nash CUF. Consider the following class of deals: A deal

$\delta = (A, A')$ is called a *Nash deal* if $\prod_{i \in \mathcal{A}^\delta} u_i(A) < \prod_{i \in \mathcal{A}^\delta} u_i(A')$. That is, a Nash deal is a deal that locally increases the Nash CUF; it increases the Nash CUF of those involved in the deal. We obtain the following convergence theorem.

Theorem 3. *Starting from any initial allocation, any sequence of Nash deals will eventually lead to a Nash-optimal allocation.*

Proof. First, it is not difficult to verify that a deal is a Nash deal if and only if it increases Nash social welfare. The claim then follows from the fact that the set of possible allocations is finite: Nash social welfare must strictly increase with each deal, so an optimal allocation will be reached eventually. $\square$

The theorem shows that any sequence of Nash deals is guaranteed to culminate in an optimal outcome and the process of negotiation cannot get stuck in a local optimum. This is particularly interesting since agents can choose the deals only depending on the effect that particular deal has within their local group. On the other hand, it must be recognized that our notion of Nash deal is conceptually less satisfying than the corresponding notion of *myopic individual rationality* used in the literature on distributed approaches to computing allocations with maximal utilitarian social welfare [19, 8]. The drawback is that agents need to share information on their utilities (even if just locally, i.e., within the group of agents participating in a specific deal) before an individual can decide whether or not a deal is acceptable to them. Also, the mechanism is not incentive-compatible (a requirement that is difficult to meet when fairness rather than just efficiency is sought). Nevertheless, whenever agents can be assumed to be cooperative, Theorem 3 shows that finding a Nash-optimal allocation can be left to a procedure requiring no central coordination.

A relevant question that arises at this point is how complicated the deals needed in the process may be. We continue with two results on the structural complexity of the Nash deals used in the negotiation process. The first is similar to a known result from the literature [8]. It makes use of the concept of independently decomposable deals: $\delta = (A, A')$ is *independently decomposable* if there are two deals $\delta_1 = (A, A'')$ and $\delta_2 = (A'', A')$ such that $\mathcal{A}^{\delta_1} \cap \mathcal{A}^{\delta_2} = \emptyset$.

Theorem 4. *For any set of agents and resources, for any particular deal δ that is not independently decomposable there exist a choice of utility functions and an initial allocation such that δ would have to be included in any sequence of Nash deals that leads to a Nash-optimal allocation.*

The proof (omitted for lack of space) involves the construction of utility functions and a non-optimal initial allocation so that δ is the only applicable Nash deal [17]. This necessity result shows that when using Nash deals to reach a Nash-optimal allocation, it may be necessary to use *any* deal that is not independently decomposable, no matter how complex it may be, and such deals may even involve all agents and all resources.

Now, it *could* still be possible to contain the structural complexity of deals if we impose constraints on the agents' utility functions. The following result shows that even restricting utility functions to the very narrow class of modular functions is not useful to this effect. A *modular* utility function u is one for which $u(X) = \alpha^{\emptyset} + \sum_{r \in X} \alpha^r$, where $\alpha^{\emptyset} = u(\emptyset)$ and $\alpha^r = u(\{r\})$.

Theorem 5. *For any set of agents and resources, if the number of resources is not less than the number of the agents, there exist a choice of utility functions and an initial allocation such that when only Nash deals are used, a deal that involves all agents may be necessary in order to reach a Nash-optimal allocation. This holds even if all utility functions are required to be modular.*

Proof. Suppose we have a distributed negotiation problem where the number of resources are at least as as many as the agents ($m \geq n$). Let A be any allocation in which each agent i owns resource r_i, $r_i \in A(i)$. We will use three parameters M, d, and ϵ in defining the utilities. Suppose that $0 < \epsilon < d < M$. We shall show that it is always possible to define these parameters such that a deal involving all agents would be necessary to reach the Nash optimal allocation.

The utility functions in modular form are defined such that $\alpha_i^{r_i} = d - \epsilon$ for all $1 \leq i \leq n$, $\alpha_i^{r_{i-1}} = M - \epsilon$ for all $1 < i \leq n$, $\alpha_1^{r_n} = M - \epsilon$, and $\alpha_i^{\emptyset} = \epsilon$ for all i. This means that each agent gives utility d to allocation A, M to allocation A' that can be reached from A if agent i gives the resource r_i to agent $(i \bmod n) + 1$ and ϵ to cases where they have neither of their two desired items.

So we have $sw_N(A) = d^n$ and $sw_N(A') = M^n$. Now for any deal starting from A in which the agents do not trade in the cycle specified above that takes A to A', the utility of the agents would not increase since either resources that are redundant in the outcome (r_i with $i > n$) would be traded, or at least one of the agents would be deprived of the resource he values d, and his utility would drop to ϵ, without the utility of another agent increasing.

Thus the only other deals we have to consider are deals that involve trading on the specified cycle, but do not complete it. Any deal of this sort of length i has the same effect. It will reduce the utility of the first agent in the chain from d to ϵ, increase the utility of the last agent to $M + d - \epsilon$, and increase the utility of the agents in between to M. In order for A' to be the only allocation that improves upon the utility of A, we need to define the parameters such that the Nash CUF is smaller than that of A in allocations resulting from all of these deals. The length of such deals can be between 1 and $n - 1$, so we must have $\epsilon M^{k-2}(M + d - \epsilon) < d^k$ for all $0 \leq k \leq n$.

It is easy to see that for any M, d and n, there exists an ϵ that satisfies

$$\epsilon < \frac{d^n}{M^{n-2}(M+d)},$$

and that it satisfies all of the other inequalities as well. So A' is the only allocation with Nash social welfare higher than A, and thus starting from A, the deal (A, A') that involves all agents would be necessary [19]. $\square$

4.3 Communication Complexity

Next, we investigate the number of deals needed to reach an optimal allocation, and prove two kinds of upper bounds on it. This aspect of the complexity of a distributed negotiation framework has been termed *communication complexity* [7] (this is different from, albeit inspired by, the use of the term in Theoretical Computer Science [24]). In our first approach to this question, we would like to know how many Nash deals are absolutely required in a sequence of deals that leads to an optimal allocation.

Theorem 6. *Starting with any initial allocation, it is always possible to reach a Nash-optimal allocation with at most one Nash deal.*

Proof. It is always possible to reach an optimal allocation with a deal from the initial allocation to the optimal one, unless the initial allocation is itself optimal, in which case no deal is needed. □

The more interesting question here does not involve the shortest negotiation sequence, but the longest one. The next result gives an upper bound on how long a negotiation process can get in the worst case, but we first need to prove the following lemma.

Lemma 1. *It is possible to define utility functions such that any two distinct allocations have different Nash social welfare.*

Proof. Let us assign to each agent i a prime number P_i so that $P_i \neq P_j$ whenever $i \neq j$. Now suppose each agent has an ordering on all possible bundles, and $u_i(R) = (P_i)^j$ if R is the j^{th} bundle in agent i's ordering. For any two distinct allocations A and A', there must be an agent k whose bundle is different in these two allocations. So the power of P_k is different in $sw_N(A)$ and $sw_N(A')$. Since any natural number has a unique prime factorization, $sw_N(A) \neq sw_N(A')$. □

Above lemma lies at the heart of the following proof. Whenever all allocations have distinct values, we can get a sequence of Nash deals of the maximal size:

Theorem 7. *Any sequence of Nash deals can consist of at most $|\mathcal{A}|^{|\mathcal{R}|} - 1$ deals.*

Proof. There are a total of $|\mathcal{A}|^{|\mathcal{R}|}$ allocations. By Lemma 1, there are utility functions for which each allocation has a different Nash CUF value. Then, if the initial allocation is the one with the least Nash CUF, and we traverse through all allocations in order of increasing Nash CUF, we obtain a sequence of $|\mathcal{A}|^{|\mathcal{R}|} - 1$ deals, all of which are Nash deals. □

5 Centralized Approach: Combinatorial Auctions

Combinatorial auctions are centralized variants of MARA. Instead of having the agents agree on an allocation interactively, they express their preferences as bids and a central authority, the *auctioneer*, computes an optimal allocation for them. In the standard case, the auctioneer is expected to compute an allocation that maximizes the utilitarian CUF (and thus revenue for the auctioneer). This is called the *winner determination problem* (WDP). Here we aim at optimizing the Nash CUF and call the corresponding problem the *Nash WDP*.

The exact definition of the WDP and the design of algorithms for solving it, both in the standard case and for our Nash combinatorial auctions, crucially depend on the *bidding language* used to encode the individual agents' preferences [15]. In this section, we shall present algorithms for solving the Nash WDP for two languages: the XOR-language and the language of positive cubes with positive weights. Theorems 1 and 2 show that both these problems are NP-hard. In the first case, we use integer programming, in the second an algorithm based on heuristic-guided search.

5.1 Winner Determination for the XOR-Language

If agent utilities are modeled using the XOR-language, then we can represent the WDP as an integer programming (IP) problem [21]. This means that, in this case, there are very powerful off-the-shelf tools available for solving the WDP. The downside of this approach is that the XOR-language is not a compact preference representation language: agents have to explicitly list all the bundles they have an interest in.

Suppose each bidder has submitted an XOR-combination of atomic bids, associating a price with a bundle of goods. Let $\langle S_{ij}, p_{ij} \rangle$ be the bundle/price pair that is the jth atomic bid of the ith bidder. We introduce a binary decision variable x_{ij} for each $\langle S_{ij}, p_{ij} \rangle$. (Index i ranges from 1 to the number of bidders; j ranges for each i from 1 to the number of atomic bids submitted by bidder i.) Solving the Nash WDP then amounts to solving the following integer program:

$$\begin{array}{l} \text{Maximize } \sum_{ij} x_{ij} \cdot \log p_{ij}, \text{ subject to} \\ (i)\ \ \sum_{j} x_{ij} \leq 1 \text{ for each bidder } i, \text{ and} \\ (ii)\ \sum \{x_{ij} \mid r \in S_{ij}\} \leq 1 \text{ for each resource } r. \end{array}$$

Constraint (i) encodes the semantics of the XOR-language and constraint (ii) ensures that each item is sold at most once. That is, we are simply using the standard IP formulation for the standard WDP with the logarithms of the prices, exploiting the relationship between utilitarian and Nash social welfare.

Unfortunately, for compact representation languages, such as the OR-language or weighted propositional formulas, this approach is not applicable. The reason is that here prices (utilities) are computed as the sum of several partial prices that a bidder has assigned to certain bundles/formulas, and the Nash CUF is the product of these sums. So the "trick" of using logarithms does not work anymore. There *are* IP solutions for the standard WDP for weighted propositional formulas [1], but it is unclear whether they could be adapted to our problem. Certainly, as Nash social welfare is defined in terms of multiplication, the Nash WDP cannot be stated directly as a linear program.

5.2 Winner Determination for the Positive Cubes Language

Here we consider the Nash WDP when the preferences are represented using the positive cubes language (with positive weights) defined in Section 2. We propose a heuristic for pruning the search tree and prove that it is admissible and also conduct some experiments and report their results.

Heuristic algorithm. Various studies have been carried out on algorithms and particularly heuristic-guided search methods for solving the WDP for classical bidding languages [9, 20]. The positive cubes language has also been considered for the standard WDP in [23]; we follow this approach for the Nash WDP.

We construct a search tree, each of its nodes being a *partial allocation* and the leaves complete allocations. A partial allocation is similar to an allocation, except that some of the items may not be assigned to any agent. The root of the tree is the partial allocation in which no resources have been assigned yet. Each child of a node corresponds to the allocation that is the result of assigning one more item to one of the agents. Thus each node has $|\mathcal{A}| = n$ children, and the depth of the tree is $|\mathcal{R}| = m$.

The algorithm is an $A^\star$ search that uses a heuristic estimate to guide search on this tree. For each node (or accordingly partial allocation) an acquired value (g) and an estimated heuristic value (h) is computed. The $A^\star$ algorithm expands the nodes in order of highest heuristic estimate value. So if the heuristic estimate of a node is less than the acquired value of another node that has already been seen it has no chance of being expanded further in the search.

The acquired value of any partial allocation is computed as the product of the acquired value of each agent, which is in turn the sum of the weights of cubes for which the agent already owns all the corresponding resources. The heuristic estimate for each partial allocation is the the product of the heuristic estimates of all the agents. Each agent's heuristic estimate is the sum of the weights of cubes for which the agent still has a chance of getting all corresponding resources, i.e., none of the resources have already been given to someone else in the partial allocation. This heuristic can be shown to be *admissible*, i.e., it never underestimates the true value of the Nash CUF of allocations that can be obtained by completing the partial allocation at any given node [17].

Theorem 8. *The function h defined above is an admissible heuristic for the Nash WDP problem when preferences are represented using the language of positive cubes with positive weights.*

Thus the heuristic-led search will always yield a Nash-optimal allocation since it will never skip a node that can possibly lead to an optimal allocation.

Experimental results. We have implemented the presented heuristic algorithm and run some experiments to test its performance.[2]

For generating the test cases we have used a simple random algorithm. Note that it would have been more desirable to use more "realistic" test data, of the kind generated by the CATS software [11], a standard benchmark for generating test cases for combinatorial auctions. However, this was not possible because CATS does not support logic-based languages. For generating the test cases, half of each agent's goalbase formulas are generated so that each of the propositional formulas has a 20% chance of containing each of the variables. The other half of

[2] The framework of the source code used to implement the algorithm was written by Joel Uckelman, and is the same as that used to run the experiments in [23].

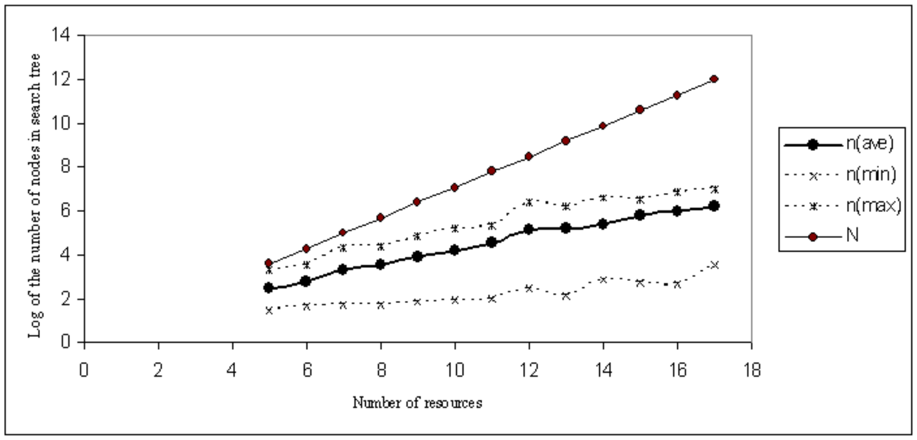

Fig. 1. The comparison of the number of nodes generated by the heuristic method and brute force search. The horizontal axis is the number of resources, the vertical axis is the logarithm in base 10 of the number of nodes of the tree generated. N is the number of total nodes of the search tree. n(ave), n(min), and n(max) are the average, minimum and maximum of the number of nodes generated in the experiments respectively.

the formulas of these goalbases each contain a single variable, these are also determined randomly. The weights are then determined by assigning each resource a nominal value and computing the weight of each bundle using that value as an estimate. This is generally appealing since it reduces the chance of having very long formulas without eliminating it altogether, and also ensures that there is a higher number of smaller formulas.

The implementation is a Java program and the experiments have been run on a Fedora Linux operating system on an Intel Pentium 4 with 3.00GHz CPU and 1 GB of RAM. It should be noted here that we have used the *best-estimate first* branching policy of [23]. This means that in each step the next good to be allocated is one of the goods that has the best estimate according to the heuristic presented in that article.

In order to assess the efficiency of the algorithm we have compared the runtime and number of nodes created in the heuristic algorithm with a brute force search that traverses all nodes of the search tree. For any number of resources between 5 and 17, we generated a setting consisting of 50 MARA scenarios containing 5 agents randomly using the method explained above. The average, minimum, and maximum number of nodes in each setting is shown in Fig. 1; the runtime of the algorithm is proportional to the number of nodes generated and is for example on average 0.58 seconds for 10 resources and 107 seconds for 17 resources. As is clear from the figure, the variation of runtimes and number of nodes generated for problems of the same size is rather high.

Although the heuristic works much better than brute force search, its runtime increases rather quickly itself. In fact, in the results for these experiments it can be observed that the runtime of the heuristic algorithm and hence the number of nodes, roughly doubles with the addition of each extra item. Note that since the runtime of the algorithm preserves a roughly constant proportion with the number of nodes even in the larger cases, the experiments do not seem to run into problems with memory, thus employing other informed search methods with the same heuristic (e.g., a branch-and-bound method) would probably not lead to significantly improved results since the $A^\star$ algorithm will provably generate a smaller search tree than any other method using the same heuristic.

It is important to keep in mind that the performance of this algorithm is not comparable to state-of-the art algorithms for combinatorial auctions. Our main purpose here has been to demonstrate that techniques from the combinatorial auction literature can in principle be applied to optimizing Nash social welfare, but this is only the first step in designing efficient algorithms in this regard.

6 Conclusion

We have presented a number of results pertaining to the quest of finding allocations that are optimal with respect to the notion of Nash social welfare, similar to those that have been obtained for other kinds of efficiency and fairness criteria in different parts of the literature on multiagent resource allocation.

First, we have shown that this is an NP-hard optimization problems, for several languages for encoding the problem. Second, in the context of a well-studied distributed negotiation framework, we have obtained a simple convergence result, and we have relativized this result by showing that deals of high structural complexity may be required during convergence and that the number of deals in a sequence leading to the optimum may be exponential. Third, we have shown that it can be instructive to view the problem of finding a Nash-optimal allocation as a variant of the winner determination problem in combinatorial auctions. We have presented two algorithms, for different preference representation languages: one using integer programming (for a en explicit representation of agent utilities) and one using heuristic-guided search (for a more sophisticated compact representation of the problem input). Finally, we have reported on initial experiments with the latter algorithm. While our experiments clearly show that there is still room for improvement, we believe to have been able to demonstrate that Nash social welfare is not only an important quality criterion, but that it is also a criterion that can be tackled successfully with the variety of methods that have been developed in the multiagent resource allocation community.

Acknowledgement. We would like to thank Joel Uckelman for providing us with the source code that facilitated the implementation of the heuristic algorithm.

References

1. C. Boutilier. Solving concisely expressed combinatorial auction problems. In *Proc. 18th National Conference on Artificial Intelligence (AAAI-2002)*, 2002.

2. S. Bouveret and J. Lang. Efficiency and envy-freeness in fair division of indivisible goods: Logical representation and complexity. *Journal of Artificial Intelligence Research*, 32:525–564, 2008.
3. Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J. A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica*, 30:3–31, 2006.
4. Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet. Reaching envy-free states in distributed negotiation settings. In *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*, 2007.
5. 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(1):49–62, 2008.
6. P. Cramton, Y. Shoham, and R. Steinberg, editors. *Combinatorial Auctions*. MIT Press, 2006.
7. U. Endriss and N. Maudet. On the communication complexity of multilateral trading. *Journal of Autonomous Agents and Multiagent Systems*, 11(1):91–107, 2005.
8. U. Endriss, N. Maudet, F. Sadri, and F. Toni. Negotiating socially optimal allocations of resources. *Journal of Artificial Intelligence Research*, 25:315–348, 2006.
9. Y. Fujishima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI-1999)*, 1999.
10. M. Grabisch. k-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, 92:167–189, 1997.
11. K. Leyton-Brown and Y. Shoham. A test suite for combinatorial auctions. In P. Cramton et al., editors, *Combinatorial Auctions*. MIT Press, 2006.
12. R. J. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *Proc. 5th ACM Conference on Electronic Commerce*, 2004.
13. H. Moulin. *Fair Division and Collective Welfare*. MIT Press, 2003.
14. J. Nash. The bargaining problem. *Econometrica*, 18(2):155–162, 1950.
15. N. Nisan. Bidding languages for combinatorial auctions. In P. Cramton et al., editors, *Combinatorial Auctions*. MIT Press, 2006.
16. C. H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
17. S. Ramezani. Nash social welfare in multiagent resource allocation. Master's thesis, ILLC, University of Amsterdam, 2008.
18. M. Rothkopf, A. Pekeč, and R. Harstad. Computationally manageable combinational auctions. *Management Science*, 44(8):1131–1147, 1998.
19. T. W. Sandholm. Contract types for satisficing task allocation: I Theoretical results. In *Proc. AAAI Spring Symposium: Satisficing Models*, 1998.
20. T. W. Sandholm. Algorithms for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1–2):1–54, 2002.
21. A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1986.
22. J. Uckelman, Y. Chevaleyre, U. Endriss, and J. Lang. Representing utility functions via weighted goals. *Mathematical Logic Quarterly*, 2009. In press.
23. J. Uckelman and U. Endriss. Winner determination in combinatorial auctions with logic-based bidding languages. In *Proc. 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2008)*, 2008.
24. A. C.-C. Yao. Some complexity questions related to distributive computing. In *Proc. 11th Annual ACM Symposium on Theory of Computing (STOC-1979)*, 1979.