Computing stable outcomes in hedonic games with voting-based deviations

Martin Gairing
Department of Computer Science
University of Liverpool
m.gairing@liverpool.ac.uk

Rahul Savani
Department of Computer Science
University of Liverpool
rahul.savani@liverpool.ac.uk

ABSTRACT

We study the computational complexity of finding stable outcomes in hedonic games, which are a class of coalition formation games. We restrict our attention to a nontrivial subclass of such games, which are guaranteed to possess stable outcomes, i.e., the set of symmetric additively-separable hedonic games. These games are specified by an undirected edge-weighted graph: nodes are players, an outcome of the game is a partition of the nodes into coalitions, and the utility of a node is the sum of incident edge weights in the same coalition. We consider several stability requirements defined in the literature. These are based on restricting feasible player deviations, for example, by giving existing coalition members veto power. We extend these restrictions by considering more general forms of preference aggregation for coalition members. In particular, we consider voting schemes to decide if coalition members will allow a player to enter or leave their coalition. For all of the stability requirements we consider, the existence of a stable outcome is guaranteed by a potential function argument, and local improvements will converge to a stable outcome. We provide an almost complete characterization of these games in terms of the tractability of computing such stable outcomes. Our findings comprise positive results in the form of polynomial-time algorithms, and negative (PLS-completeness) results. The negative results extend to more general hedonic games.

Keywords

Hedonic games, coalition formation, voting, local search, PLS-completeness.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; F.2.0 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms

Algorithms, Economics, Theory

Cite as: Computing stable outcomes in hedonic games with voting-based deviations, M. Gairing and R. Savani, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. 559-566.

1. INTRODUCTION

Hedonic games were introduced in the economics literature as a model of coalition formation where each player cares only about those within the same coalition [12]. Such games can be used to model a variety of settings ranging from multi-agent coordination to group formation in social networks. This paper studies the computational complexity of finding stable outcomes in hedonic games. We consider and extend the stability requirements introduced in the work of Bogomolnaia and Jackson [6], which includes a detailed discussion of real-life situations in which purely hedonic models are reasonable.

An outcome is called *Nash-stable* if no player prefers to be in a different coalition. Here a deviation depends only on the preferences of the deviating player. Less stringent stability requirements restrict feasible deviations: a coalition may try to hold on to an attractive player or block the entry of an unattractive player. In [6], deviations are restricted by allowing members of a coalition to "veto" the entry or exit of a player. They introduce *individual stability*, where there is a veto for entering - a player can deviate to another coalition only if *everyone* in this coalition is happy to have her. They also introduce *contractual individual stability*, where, in addition to a veto for entering, coalition members have a veto to prevent a player from leaving the coalition - a player can deviate only if everyone in her coalition is happy for her to leave.

The case where every member of a coalition has a veto on allowing players to enter and/or leave the coalition can be seen as an extreme form of *voting*. This motivates the study of more general voting mechanisms for allowing players to enter and leave coalitions. In this paper, we consider general voting schemes, for example, where a player is allowed to join a coalition if the majority of existing members would like the player to join. We also consider other methods of *preference aggregation* for coalition members. For example, a player is allowed to join a coalition only if the aggregate utility (i.e., the sum of utilities) existing members have for the entrant is non-negative. These preference aggregation methods are also considered in the context of preventing a player from leaving a coalition. We study the computational complexity of finding stable outcomes under stability requirements with various restrictions on deviations.

The model.

In this paper, we study hedonic games with *symmetric additively-separable* utilities, which allow a succinct representation of the game as an *undirected edge-weighted graph*

$G = (V, E, w)$. For clarity of our voting definitions, we assume w.l.o.g. that $w_e \neq 0$ for all $e \in E$. Every node $i \in V$ represents a player. An outcome is a partition p of V into coalitions. Denote by $p(i)$ the coalition to which $i \in V$ belongs under p, and by $E(p(i))$ the set of edges $\{\{i, j\} \in E \mid j \in p(i)\}$.

The utility of $i \in V$ under p is the sum of edges to others in the same coalition, i.e., $\sum_{e \in E(p(i))} w(e)$. Each player wants to maximize her utility, so a player *wants to deviate* if there exists a (possibly empty) coalition c where

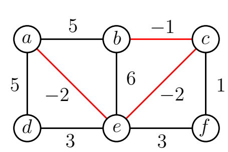
$$\sum_{e \in E(p(i))} w(e) < \sum_{\{\{i,j\} \in E \ \mid \ j \in c\}} w(\{i, j\}).$$

We consider different restrictions on player deviations. Those restrict when players are allowed to join and/or leave coalitions. A deviation of player i to coalition c is called

- *Nash feasible* if player i wants to deviate to c.
- *vote-in feasible* with threshold T_{in} if it is Nash feasible and either at least a T_{in} fraction of i's edges to c are positive or i has no edge to c.
- *vote-out feasible* with threshold T_{out} if it is Nash-feasible and either at least a T_{out} fraction of i's edges to $p(i)$ are negative or i has no edges within $p(i)$.
- *sum-in feasible* if it is Nash feasible and
$$\sum_{\{\{i,j\} \in E \ \mid \ j \in c\}} w(\{i, j\}) \geq 0.$$
- *sum-out feasible* if it is Nash feasible and
$$\sum_{e \in E(p(i))} w(e) \leq 0.$$

Outcomes where no corresponding feasible deviation is possible are called *Nash stable, vote-in stable, vote-out stable, sum-in stable*, and *sum-out stable*, respectively. Outcomes which are vote-in (resp. vote-out) stable with $T_{in} = 1$ (resp. $T_{out} = 1$) are also called *veto-in* (resp. *veto-out*) *stable*. Note that a veto-in stable outcome is an *individual stable* outcome i.e., any player can veto a player joining a coalition; an outcome that is veto-in and veto-out stable is a *contractual individual stable* outcome.

An example.



The above figure gives an example of a hedonic game. Consider the outcome $\{\{a, b, d\}, \{c, e, f\}\}$. The utilities of the players a, b, c, d, e, f are $10, 5, -1, 5, 1, 4$, respectively. Players a, b, d, f have no Nash-feasible deviations, c has a Nash-feasible deviation to go alone and start a singleton coalition, and e has a Nash-feasible deviation to join the other coalition. The deviation of c is not veto-out feasible, since f prefers c to stay, however it is vote-out feasible for any $T_{out} \leq 0.5$. It is also sum-out feasible. The deviation of e is not veto-in feasible, but is vote-in feasible for any $T_{in} \leq 2/3$. Since there are no deviations that are both veto-in and veto-out feasible, this is a contractual individual stable outcome. The outcome $\{\{a, b, d\}, \{c\}, \{e, f\}\}$ is an individual stable outcome, and $\{\{a, b, d, e, f\}, \{c\}\}$ is Nash stable.

Justification of the model.

With the goal of understanding how difficult it is for agents to *find* stable outcomes, we focus on a model in which they are *guaranteed* to exist. The computational complexity of a problem is measured in terms of the size of its input and therefore depends on the representation of the problem instance. For games, we desire that the size of the input is polynomial in the number of players, as this is the natural parameter with which to measure the size of the game. We consider only such *succinct representations*, since otherwise we can find solutions using trivial algorithms (enumeration of strategy profiles) that are polynomial in the input size. Our focus on additively-separable games is motivated by the hardness of even deciding the existence of stable outcomes and other solution concepts for more general (universal) succinct representations, such as hedonic nets [14]. A *non-symmetric* additively-separable game, which is represented by a edge-weighted *directed* graph, may not have a Nash-stable outcome [6, 4], and deciding existence is NP-complete. We study a more restrictive model where stable outcomes (for all of the stability requirements we consider) are guaranteed to exist, noting that our hardness results extend to all more general models where existence of stable outcomes is either guaranteed or promised, i.e., instances are restricted to those possessing stable outcomes.

In a *symmetric* additively-separable hedonic game, for each of the stability requirements we consider, a stable outcome *always exists* by a simple potential function argument: the potential function is the total happiness of an outcome, i.e., the sum of players' utilities. Unilateral player deviations improve the potential. So for all our considered stability requirements, local improvements will find a stable outcome, and all the problems we consider are in the complexity class PLS (polynomial local search) [20], which we introduce next.

Local search and the complexity class PLS.

Local search is one of few general and successful approaches to difficult combinatorial optimisation problems. A local search algorithm tries to find an improved solution in the *neighborhood* of the current solution. A solution is *locally optimal* if there is no better solution in its neighborhood. Johnson et al. [20] introduced the complexity class PLS (polynomial local search) to capture those local search problems for which a better neighboring solution can be found in polynomial time if one exists, and a local optimum can be verified in polynomial time.

They also introduced the notion of *PLS-reduction*. Suppose A and B are problems in PLS. Then A is PLS-reducible to B if there exist polynomial time computable functions f and g such that f maps instances of A to instances of B and g maps the local optima of B to local optima of A. A problem is PLS-complete if all problems in PLS are PLS-reducible to it. Prominent PLS-complete problems are those of finding a local max-cut in a graph (LocalMaxCut) [24], a stable solution in a Hopfield network [20], or a pure Nash equilibrium in a congestion game [16]. PLS captures the problem of finding pure Nash equilibria for many classes of games where pure equilibria are guaranteed to exist.

On the one hand, finding a locally optimal solution is presumably easier than finding a global optimum; in fact, it is very unlikely that a PLS problem is NP-hard since this would imply NP=coNP [20]. On the other hand, a polynomial-time algorithm for a PLS-complete problem would immediately imply such an algorithm for all problems in PLS and thus solve a number of long open problems including the *simple stochastic game problem* [29]. PLS-complete problems are believed not to have polynomial-time algorithms.

Computational problems.

We define the search problems, NashStable, IS (individual stable), CIS (contractual individual stable), VoteIn, and VoteOut of finding a stable outcome for the respective stability requirement. We introduce VoteInOut as the search problem of finding an outcome which is vote-in and vote-out stable. All voting problems are parametrized by T_{in} and/or T_{out}. We also introduce sumCIS as the problem of finding an outcome which is sum-in and sum-out stable.

Symmetric additively-separable hedonic games are closely related to party affiliation games, which are also specified by an undirected edge-weighted graph. In a party affiliation game each player must choose between one of two "parties"; a player's happiness is the sum of her edges to nodes in the same party; in a stable outcome no player would prefer to be in the other party. The problem PartyAffiliation is to find a stable outcome in such a game. If such an instance has only negative edges then it is equivalent to the problem LocalMaxCut, which is to find a stable outcome of a local max-cut game. In party affiliation games there are at most two coalitions, while in hedonic games any number of coalitions is allowed. Thus, whereas PartyAffiliation for instances with only negative edges is PLS-complete [24], NashStable is trivial in this case, as the outcome where all players are in singleton coalitions is Nash-stable. Both problems are trivial when all edges are non-negative, in which case the grand coalition of all players is Nash-stable. Thus, interesting hedonic games contain both positive and negative edges.

The problem OneEnemyPartyAffiliation is to find a stable outcome of a party affiliation game where each node is incident to at most one negative edge. This problem was introduced in [17]. In this paper, we use a variant of this problem as a starting point for some of our reductions:

Definition 1. *Define the problem* OneEnemyPartyAffiliation* *as a restricted version of* OneEnemyPartyAffiliation *which is restricted to instances where no player is ever indifferent between the two coalitions.*

Gairing and Savani [17, Corollary 1] showed that OneEnemyPartyAffiliation* is PLS-complete.

Our results.

In this paper, we examine the complexity of computing stable outcomes in symmetric additively-separable hedonic games. In [17], it was shown that NashStable is PLS-complete while CIS is solvable in polynomial time. We make explicit two conditions, both met in the case of CIS, that (individually) guarantee that local improvements converge in polynomial time. The complexity of IS (i.e., of finding a veto-in stable outcome) was left open in [17]. Here we resolve that question, showing that IS is PLS-complete. Perhaps surprisingly, given the apparently restrictive nature of the stability requirement, we show that sumCIS is PLS-complete, in contrast to CIS.

We also study the complexity of finding vote-in and vote-out stable outcomes. Using a different argument to the polynomial-time cases mentioned previously, we show that local improvements converge in polynomial time in the case of vote-in- and vote-out- stability with $T_{in}, T_{out} \geq 0.5$ and $T_{in} + T_{out} > 1$. We show that if we require vote-in-stability alone, we get a PLS-complete search problem. The problem of finding a vote-out stable outcome is conceptually different, and we can find a veto-out-stable outcome in polynomial time (whereas it is PLS-complete to find a veto-in-stable outcome). The technical difficulty in proving a hardness result for VoteOut is restricting the number of coalitions. Ultimately, we leave the complexity of VoteOut open, but do show that k-VoteOut, which is the problem of computing a vote-out stable outcome when at most k coalitions are allowed, is PLS-complete (Theorem 2). Our results are summarized in Figure 1, which gives an almost complete characterization of tractability.

Related work.

Hedonic coalition formation games were first considered by Dreze and Greenberg [12]. Greenberg [18] later surveyed coalition structures in game theory and economics. Based on [12], Bogomolnaia and Jackson [6] formulated different stability concepts in the context of hedonic games - see also the survey [26]. These stability concepts were our motivation to introduce definitions of stability based on voting and aggregation.

The general focus in the game theory community has been on characterizing the conditions for which stable outcomes exist. Burani and Zwicker [8] showed that additively-separable and symmetric preferences guarantee the existence of a Nash-stable outcome. They also showed that under certain different conditions on the preferences, the set of Nash-stable outcomes can be empty but the set of individually-stable partitions is always non-empty.

Cechlárová [9] surveys algorithmic problems related to stable outcomes. Ballester [4] showed that for hedonic games represented by an *individually rational list of coalitions*, the complexity of checking whether core-stable, Nash-stable or individual-stable outcomes exist is NP-complete, and that every hedonic game has a contractually-individually-stable solution. Recently, Sung and Dimitrov [27] showed that for additively-separable hedonic games checking whether a core-stable, strict-core-stable, Nash-stable or individually-stable outcome exists is NP-hard. For core-stable and strict-core-stable outcomes those NP-hardness results have been extended by Aziz et al. [2] to the case of symmetric player preferences. Brânzei and Larson [7] studied the tradeoff between stability and social welfare in additively-separable hedonic games. Elkind and Wooldridge [14] characterize the complexity of problems related to coalitional stability for hedonic games represented by hedonic nets, a succinct, rule-based representation based on marginal contribution nets (introduced by Ieong and Shoham [19]).

This work extends the model and results in Gairing and Savani [17]. The definition of party affiliation games we use appears in Balcan et al. [3]. Recent work on local max cut and party affiliation games has focused on approximation [5, 10]; see also [23]. For surveys on the computational

Leave \ Enter	1: no restr.	2: sum-in	3: veto-in	4: vote-in
A: no restr.	NashStable PLS-complete [17]	PLS-complete [17]	IS PLS-complete Theorem 4	VoteIn PLS-complete Theorem 1
B: sum-out	PLS-complete Theorem 5	sumCIS PLS-complete Theorem 5	P Proposition 1	?
C: veto-out	P Proposition 2	P Proposition 2	CIS P [17]	P Proposition 2
D: vote-out	VoteOut ? (see Theorem 2)	? (see Theorem 2)	P Proposition 1	VoteInOut P ($T_{in}, T_{out} > 0.5$) Theorem 3

Figure 1: Table showing the computational complexity of the search problems for different entering and leaving deviation restrictions. Note that columns 1 and 2 are essentially equivalent, since if a player has a Nash-feasible deviation that results in a negative payoff, she also has a sum-in feasible (and hence also Nash-feasible) deviation, namely to form a singleton coalition.

complexity of local search, see [22, 1]. We use the PLS-completeness of LocalMaxCut which was shown in Schäffer and Yannakakis [24].

There is an extensive literature on weighted voting games, which are formally simple coalitional games. For such a game, a "solution" is typically a vector (or set of vectors) of payoffs for the players, rather than a coalition structure as in our setting; for recent work on computational problems associated with weighted voting games see [13, 15]. Deng and Papadimitriou [11] examined the computational complexity of computing solutions for coalitional games for a model similar to additively-separable hedonic games, where the game is given by an edge-weighted graph, and the value of a coalition of nodes is the sum of weights of edges in the corresponding subgraph. Here, we study the complexity of finding a stable set of coalitions.

2. COMPUTATIONAL COMPLEXITY OF FINDING STABLE OUTCOMES

In this section we study the complexity of computing stable outcomes under various stability requirements. We start by showing PLS-hardness for the case that a deviating player needs a T_{in} majority in the target coalition but there is no restriction on leaving coalitions.

Theorem 1. VoteIn *is PLS-complete for any voting threshold* $0 \leq T_{in} < 1$.

Proof. We reduce from OneEnemyPartyAffiliation* represented by an edge-weighted graph $G = (V, E, w)$. Let $\Delta(G)$ be the maximum degree of a node in G. Recall that no player is ever indifferent between the two coalitions.

First observe that the case $T_{in} > \frac{\Delta(G)-1}{\Delta(G)}$ is exactly the same as IS (for which we show hardness in Theorem 4), since in this case one negative edge is enough to veto a player joining a coalition. In the following we assume $T_{in} \leq \frac{\Delta(G)-1}{\Delta(G)}$.

We augment G as follows:

For every negative edge (a, b) in G we introduce $2\Delta(G)-2$ new nodes, called *followers*, and connect them with a and b as shown in the Figure 2. Both, a and b, get $\Delta(G) - 1$ followers and have a δ edge to each of them. Moreover, the followers have also an edge of weight ε to the other node.

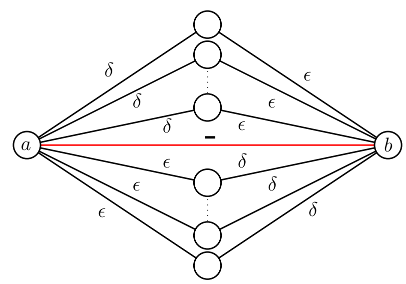


Figure 2: Gadget used for showing that VoteIn is PLS-complete. The gadget augments negative edges with followers that ensure that there is always a T_{in}-majority when a player enters a coalition.

Here $0 < \varepsilon < \delta$ and δ is small enough so that the player preferences of the original players (a and b) are still determined only by the original edges. In a stable outcome the followers will be in the same coalition as their "leader", i.e., the node to which they have a δ edge. The followers make sure that their is always a T_{in}-majority for entering a coalition. In other words, in a stable outcome of the VoteIn instance, the voting doesn't impose any restrictions.

To ensure that any stable outcome for the VoteIn instance has only two coalitions we further augment G by introducing two new players, called *supernodes*. Every player $i \in V$ has an edge of weight $W > \sum_{e \in E} |w_e|$ to each of the supernodes. The two supernodes are connected by an edge of weight $-M$, where $M > |V| \cdot W$. This enforces that the two supernodes are in a different coalition in any stable outcome. Moreover, by the choice of W, each player in V will be in a coalition with one of the supernodes. The fact that edges to supernodes have all the same weight directly implies that a stable outcome for the VoteIn instance is also a stable outcome for the OneEnemyPartyAffiliation* instance. The claim follows. $\square$

In contrast to VoteIn, VoteOut is conceptually different. In VoteOut a coalition of two players connected by a positive edge is vote-out stable. This makes it hard to restrict the number of coaltions. Doing this is probably the key for proving PLS-hardness also for VoteOut. For the following theorem we consider a version of VoteOut where the number of coalitions are restricted by the problem. Let k-VoteOut be the problem of computing a vote-out stable outcome when at most k coalitions are allowed. Observe that for any $k \geq 2$ such a vote-out stable outcome exists and that local improvements starting from any k-partition converge to such a stable outcome.

Theorem 2. *k-VoteOut is PLS-complete for any voting threshold $0 \leq T_{out} < 1$ and any $k \geq 2$.*

Proof. Our reduction is from OneEnemyPartyAffiliation, but we first reduce to the intermediate problem OneEnemyNashStable, which is a restricted version of NashStable where each player is only incident to at most one negative edge. Consider an instance of OneEnemyPartyAffiliation which is represented as an edge-weighted graph $G = (V, E, w)$. We augment G with two supernodes in exactly the same way as in Theorem 1. This ensures that any stable outcome of the OneEnemyNashStable instance uses only two coalitions and thus is also a stable outcome for the OneEnemyPartyAffiliation instance. Hence, OneEnemyNashStable is PLS-complete.

We now reduce from OneEnemyNashStable to k-VoteOut. Let G be the graph corresponding to an instance of OneEnemyNashStable. Let $\Delta(G)$ be the maximum degree of a node in G. We augment G as follows: We introduce $s \cdot k \cdot \Delta(G)$ new nodes where s is an integer satisfying $s \geq \frac{T_{out}}{1-T_{out}}$. Those nodes are organized in $s \cdot \Delta(G)$ complete graphs of k nodes each. All the edges in the complete graphs have weight $-M$ where M is sufficiently large ($M > |V| \cdot \Delta(G) \cdot \varepsilon$ will do). Moreover, we connect every original node $u \in V$ to every new node with an edge of weight $-\varepsilon$, where $\varepsilon > 0$.

By the choice of M and since at most k coalitions are allowed, in any stable solution there will be one node from each complete graph in each of the k coalitions. This shifts the utility of each player $i \in V$ with respect to each coalition by $-s \cdot \Delta(G) \cdot \varepsilon$. Moreover, every original node has at least $s \cdot \Delta(G)$ negative edges to each coalition. Since each node is incident to at most $\Delta(G)$ positive edges, it follows that the fraction of negative edges to each coalition is at least $\frac{s}{s+1} \geq T_{out}$. Thus, in every stable outcome all nodes $u \in V$ have a T_{out}-majority for leaving their coalition. This implies that in the corresponding outcome of the OneEnemyNashStable instance, no player can improve her utility by joining one of the k coalitions used in k-VoteOut. Moreover, in every stable outcome the utility of each node $u \in V$ with respect to the set of original nodes V is non-negative, since u has at most one negative incident edge in the OneEnemyNashStable instance and $k \geq 2$. It follows that a stable outcome for the k-VoteOut instance is also a stable outcome for the OneEnemyNashStable instance. The claim follows. $\square$

It is an interesting open problem whether PLS-completeness also holds if the restriction on the number of allowed coalitions is dropped. Can we construct a gadget that imposes this restriction without restricting the problem a priori?

Since VoteIn and a restricted version of VoteOut are PLS-complete it's interesting to study the combination of both problems. What happens if we require vote-in stability *and* vote-out stability? With a mild assumption on the voting thresholds T_{in}, T_{out}, we establish:

Theorem 3. *For any instance of VoteInOut with voting thresholds $T_{in}, T_{out} \geq \frac{1}{2}$ and $T_{in} + T_{out} > 1$, local improvements converge in $\mathcal{O}(|E|)$ steps.*

Proof. For any outcome p define a potential function $\Phi(p) = \Phi^+(p) - \Phi^-(p)$, where $\Phi^+(p)$ (resp. $\Phi^-(p)$) is the number of positive (resp. negative) internal edges, i.e. edges not crossing coalition boundaries. Consider a local improvement of some player i from coalition $p(i)$ to $p'(i)$. Since $T_{out} \geq \frac{1}{2}$, player i has at least as many negative as positive edges to $p(i)$. Likewise since $T_{in} \geq \frac{1}{2}$, player i has at least as many positive as negative edges to $p'(i)$. So $\Phi(p)$ cannot decrease by a local improvement. Moreover, since $T_{in} + T_{out} > 1$, one of the threshold inequalities must be strict, which implies $\Phi(p') > \Phi(p)$. The claim follows since $-|E| \leq \Phi(p) \leq |E|$ and $\Phi(p)$ is integer. $\square$

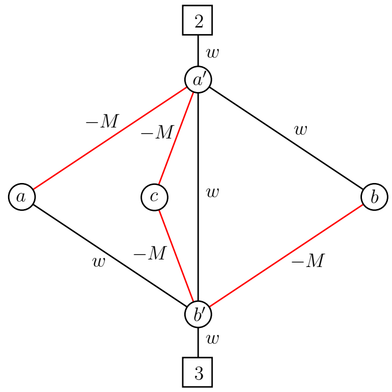
Without the assumption on the voting thresholds, the complexity of computing stable outcomes remains an interesting open problem. In particular the case $T_{in} = T_{out} = 1/2$ is very tantalizing.

We proceed by studying the complexity of finding stable outcomes if a single player in the target coalition can prevent (veto) a player from joining it. Observe, that the proof of Theorem 1 does not go through for this case. In [17] it was shown that a restricted version of IS (where in addition to normal IS deviations, two players connected by an negative edge are allowed to swap coalitions) is PLS-complete. Here, we show that allowing swaps is not necessary for PLS-hardness.

Theorem 4. IS *is PLS-complete.*

Proof. We start with an instance of OneEnemyPartyAffiliation*. The instance has the property that no player is ever indifferent between the two coalitions that make up stable outcomes. We add four supernodes which are connected by a complete graph of sufficiently large negative edges. This enforces that in any stable outcome the supernodes are in different coalitions, say 0, 1, 2, 3. The supernodes are used to restrict which coalition a node can be in in a stable outcome. This is achieved by having large positive edges of equal weight to the corresponding supernodes. All original nodes of the OneEnemyPartyAffiliation* instance are restricted to be 0 or 1.

We now show how to simulate a negative edge of OneEnemyPartyAffiliation* by an IS-gadget. To do so, we replace a negative edge (a, b) of weight $-w$ with the gadget in Figure 3. Nodes a and b are original nodes and restricted to $\{0, 1\}$, node a' is restricted to $\{0, 1, 2\}$, node b' is restricted to $\{0, 1, 3\}$, and node c is restricted to $\{2, 3\}$. As depicted in the gadget, nodes a' and b' have an additional offset to 2 and 3, respectively. Coalitions 2 and 3 are only used locally within the gadget. The pseudocode next to the gadget describes how the internal nodes of the gadget are biased. Here, checking whether a node can improve is w.r.t. her *original* neighborhood. We use "look at" and "bias" as defined in the following lemma and definition, which are analogous to those in [28, 21]. In particular, we check if a node can improve by looking at all nodes in her original neighborhood.



Bias internal nodes

if a can improve **then**
    bias c to 3
    bias a' to 2
else
    bias a' to $\{0,1\}$
    bias c to 2
end if
if b can improve **then**
    bias b' to 3
else
    bias b' to $\{0,1\}$
end if

Figure 3: Gadget to replace negative edges

LEMMA 1. *For any polynomial-time computable function $f : \{0,1\}^k \mapsto \{0,1,2,3\}^m$ one can construct a graph $G_f = (V_f, E_f, w)$ having the following properties: (i) there exist $s_1, \ldots, s_k$, $t_1, \ldots, t_m \in V_f$, (ii) all edges $e \in E_f$ are positive, (iii) $f(s_1, \ldots, s_k) = (t_1, \ldots, t_m)$ in any stable solution of the hedonic game defined by G_f.*

DEFINITION 2. *For a polynomial-time computable function $f : \{0,1\}^k \mapsto \{0,1,2,3\}^m$ we say that G_f as constructed in Lemma 1 is a graph that* looks at *$s_1, \ldots, s_k \in V_f$ and* biases *$t_1, \ldots, t_m \in V_f$ according to the function f.*

Recall that the instance of ONEENEMYPARTYAFFILIATION* has the property that no player is ever indifferent between the two coalitions that make up stable outcomes. By scaling edge weights we can implement the "look at" required to bias the internal nodes of the gadget without affecting their original preferences.

We say that node a is *locked* by the gadget if $a = 1$ and $a' = 0$ or $a = 0$ and $a' = 1$. Node b is said to be locked accordingly. The following two lemmas describe the operation of the gadget. Both lemmas should be read with the implicit clause: *If the internal nodes (a', b', c) are stable.* Let $\neg u$ denote the complement of u over $\{0,1\}$.

LEMMA 2. *If neither a nor b can improve then a and b are locked by the gadget.*

LEMMA 3. *If a or b (or both) can improve then one improving node is not locked while the other node is locked by the gadget. Moreover, if a (resp. b) is not locked by the gadget then $b' = \neg b$ (resp. $a' = \neg a$).*

To complete the proof we show that a stable outcome of the IS instance is also a stable outcome for the ONEENEMYPARTYAFFILIATION* instance. Suppose the contrary. Then there must exist an original node which is stable for IS but not for ONEENEMYPARTYAFFILIATION*. Clearly such a node must be the node a or b for some gadget. So either a or b (or both) can improve. But then by the first statement in Lemma 3 one of the improving nodes is unlocked, say a. Since a was only incident to one negative edge in the ONEENEMYPARTYAFFILIATION* instance, a cannot be locked by any other gadget. Moreover, by the second statement in Lemma 3, a is now connected in the gadget by a positive edge to the node b' and $b' = \neg b$. On the one hand, if $a = b$ then the original edge (a, b) contributes $-w$ to a's utility while now a receives 0 from the edge (a, b'). On the other hand, if $a \neq b$ then the corresponding utility contributions are 0 and w. So if a changes strategy then the difference in her utility w.r.t. b is the same in both problems, since we just shifted the utility of node a w.r.t. b by w. So a is also not stable for IS, a contradiction. This finishes the proof of Theorem 4. $\square$

In IS a single player can veto against others joining her coalition but there is no restriction on leaving a coalition. The following proposition shows that adding certain leaving conditions yields polynomial-time convergence from the all-singleton partition.

PROPOSITION 1. *Any problem in column 3 of Figure 1 can be solved in polynomial time provided that the leaving condition requires that the leaving node has at least one negative edge within the coalition. In particular this hold for the problems in cells 3B, 3C, and 3D.*

PROOF. We use local improvements starting from the set of singleton coalitions. Then a player can make at most one improving step, since all edges in resulting non-singleton coalitions will be positive, and so no player can leave such a coalition. Hence we arrive at a stable outcome in at most $|V|$ improving steps. $\square$

Interestingly, requiring veto-feasablity is already enough for polynomial-time convergence even if we have no restriction on the entering condition. This stands in contrast to Theorem 4.

PROPOSITION 2. *All problems in row C of Figure 1 can be solved in polynomial time by local improvements using at most $2|V|$ improving steps.*

PROOF. To get a running time of $2|V|$ (rather than $O(|V|^2)$) we restrict players from joining a non-empty coalition to which they have no positive edge. This ensures that whenever a player joins a non-empty coalition then this player (and all players to which she is connected by a positive edge in the coalition) will never move again. Moreover, a player can only start a new coalition once. It follows that each player can make at most two strategy changes. In total we have at most $2|V|$ local improvements. $\square$

We close this paper with a result for SUMCIS. Even though deviations are very restricted here, it is PLS-complete to compute a stable outcome.

THEOREM 5. SUMCIS *is PLS-complete.*

PROOF. We reduce from LOCALMAXCUT. Consider an arbitrary instance of LOCALMAXCUT with only integer edge weights. Recall that such an instance can be cast as an instance of PARTYAFFILIATION by negating the weights of the edges. Let $G = (V, E, w)$ represent the PARTYAFFILIATION instance. For each player $i \in V$ let σ_i be the total weight of edges incident to player i, i.e. $\sigma_i = \sum_{(i,j)\in E} w_{(i,j)}$. Observe that σ_i is a negative integer. We augment G by introducing two new players, called *supernodes*. Every player $i \in V$ has an edge of weight $\frac{-\sigma_i}{2} + \frac{1}{4}$ to each supernode. The two supernodes are connected by an edge of weight $-M$ where M is sufficiently large (i.e., $M > \sum_{i\in V}(\frac{-\sigma_i}{2} + \frac{1}{4})$). The resulting graph G' represents our SUMCIS instance.

Consider a stable outcome of the SUMCIS instance G'. By the choice of M the two supernodes will be in different coalitions. Now consider any player $i \in V$. If i is not in a coalition with one of the supernodes, then i's payoff is negative. On the other hand joining the coalition of one of the supernodes yields positive payoff, since $2(\frac{-\sigma_i}{2} + \frac{1}{4}) + \sigma_i > 0$. Thus, each player $i \in V$ will be in a coalition with one of the supernodes. So our outcome partitions V into two partitions, say V_1, V_2.

It remains to show that any stable outcome for the SUMCIS instance is also a local optimum for the PARTYAFFILIATION instance. Assume that the outcome of the SUMCIS instance is stable but in the corresponding outcome of PARTYAFFILIATION instance there exists a player i which can improve by joining the other coalition. W.l.o.g. assume $i \in V_1$. Then, $\sum_{s\in V_1} w_{(i,s)} < \sum_{s\in V_2} w_{(i,s)}$. With $\sigma_i = \sum_{s\in V} w_{(i,s)}$ and since σ_i is integer, we get

$$\sum_{s\in V_1} w_{(i,s)} \leq \frac{\sigma_i}{2} - \frac{1}{2} < \frac{\sigma_i}{2} < \frac{\sigma_i}{2} + \frac{1}{2} \leq \sum_{s\in V_2} w_{(i,s)}.$$

It follows that in the SUMCIS instance, player i's payoff is negative in her current coalition V_1 whereas joining V_2 would yield positive payoff. This contradicts our assumption that we are in a stable outcome of the SUMCIS instance. The claim follows. □

3. CONCLUSIONS AND OPEN PROBLEMS

Our findings comprise both positive and negative results, some of which are somewhat surprising. There is an asymmetry between the case of vote-in and vote-out stability. We show that VOTEIN is PLS-complete for all voting thresholds, including $T_{in} = 1$. The case for $T_{in} = 1$, which corresponds to the search problem IS for finding a veto-in stable outcome, has to be treated separately from the case $T_{in} < 1$. In contrast, we show that the case of finding a veto-out stable outcome is polynomial-time solvable. This suggests that VOTEOUT is conceptually different from VOTEIN. Indeed, it seems difficult to restrict the coalitions in this case. We do show that k-VOTEOUT, where we restrict the outcome to have at most k coalitions, is PLS-complete for $0 \leq T_{out} < 1$, but we leave the complexity of VOTEOUT as an interesting open problem.

We show that even though requiring both sum-in and sum-out stability is apparantly quite restrictive, the resulting search problem SUMCIS is PLS-complete.

In terms of positive results, we show that local improvements converge in polynomial time in the case of requiring *both* vote-in- and vote-out- stability with $T_{in}, T_{out} \geq 0.5$ and $T_{in} + T_{out} > 1$. We leave open the interesting case of VOTEINOUT with voting thresholds that do not satisfy $T_{in}, T_{out} \geq \frac{1}{2}$ and $T_{in} + T_{out} > 1$. We also leave open the case of finding an outcome that is vote-in and sum-out stable.

References

[1] E. H. L. Aarts and J. K. Lenstra. *Local Search in Combinatorial Optimization.* Wiley-Interscience, 1997.

[2] H. Aziz, F. Brandt, H. G. Seedig. Stable partitions in additively separable hedonic games. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2011.

[3] M.-F. Balcan, A. Blum, and Y. Mansour. Improved equilibria via public service advertising. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pp. 728–737, 2009.

[4] C. Ballester. NP-completeness in hedonic games. *Games and Economic Behavior*, 49(1):1–30, 2004.

[5] A. Bhalgat, T. Chakraborty, and S. Khanna. Approximating pure Nash equilibrium in cut, party affiliation and satisfiability games. In *ACM Conference in Electronic Commerce (EC)*, pp. 132–146, 2010.

[6] A. Bogomolnaia and M. O. Jackson. The stability of hedonic coalition structures. *Games and Economic Behavior*, 38(2):201–230, 2002.

[7] S. Brânzei and K. Larson. Coalitional affinity games and the stability gap. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 79–84, 2009.

[8] N. Burani and W. S. Zwicker. Coalition formation games with separable preferences. *Mathematical Social Sciences*, 45(1):27–52, 2003.

[9] K. Cechlárová. Stable partition problem. In *Encyclopedia of Algorithms.* Springer, 2008.

[10] G. Christodoulou, V. S. Mirrokni, and A. Sidiropoulos. Convergence and approximation in potential games. In *Symp. on Theoretical Aspects of Computer Science (STACS)*, pp. 349–360, 2006.

[11] X. Deng and C. H. Papadimitriou. On the complexity of cooperative solution concepts. *Mathematics of Operations Research*, 12(2):257–266, 1994.

[12] J. H. Dreze and J. Greenberg. Hedonic coalitions: Optimality and stability. *Econometrica*, 48(4):987–1003, 1980.

[13] E. Elkind and D. Pasechnik. Computing the nucleolus of weighted voting games. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pp. 327–335, 2009.

[14] E. Elkind and M. Wooldridge. Hedonic coalition nets. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 417–424, 2009.

[15] E. Elkind, L. A. Goldberg, P. W. Goldberg, and M. Wooldridge. On the computational complexity of weighted voting games. *Ann. Math. Artif. Intell.*, 56 (2):109–131, 2009.

[16] A. Fabrikant, C. H. Papadimitriou, and K. Talwar. The Complexity of Pure Nash Equilibria. In *ACM Symp. on Theory of Computing (STOC)*, pp. 604–612, 2004.

[17] M. Gairing and R. Savani. Computing stable outcomes in hedonic games. In *International Symp. on Algorithmic Game Theory (SAGT)*, pp. 174–185, 2010.

[18] J. Greenberg. Coalition structures. In R. J. Aumann and S. Hart, editors, *Handbook of Game Theory with Economic Applications*, volume II. Elsevier, 1994.

[19] S. Ieong and Y. Shoham. Marginal contribution nets: A compact representation scheme for coalitional games. In *ACM Conference on Electronic Commerce (EC), 2005*, pp. 193–202, 2005.

[20] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37:79–100, 1988.

[21] B. Monien and T. Tscheuschner. On the power of nodes of degree four in the local max-cut problem. In *International Conference on Algorithms and Complexity (CIAC)*, pp. 264–275, 2010.

[22] B. Monien, D. Dumrauf, and T. Tscheuschner. Local search: Simple, successful, but sometimes sluggish. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pp. 1–17 , 2010.

[23] J. B. Orlin, A. P. Punnen, and A. S. Schulz. Approximate local search in combinatorial optimization. *SIAM Journal on Computing*, 33(5):1201–1214, 2004.

[24] A. A. Schäffer and M. Yannakakis. Simple Local Search Problems that are Hard to Solve. *SIAM Journal of Computing*, 20(1):56–87, 1991.

[25] M. Sipser. *Introduction to the Theory of Computation.* Thomson, 2006.

[26] S. C. Sung and D. Dimitrov. On myopic stability concepts for hedonic games. *Theory and Decision*, 62, 2007.

[27] S. C. Sung and D. Dimitrov. Computational complexity in additive hedonic games. *European Journal of Operational Research*, 203(3):635–639, 2010.

[28] T. Tscheuschner. The local max-cut problem is PLS-complete even on graphs with maximum degree five. *http://arxiv.org/abs/1004.5329*, 2010.

[29] M. Yannakakis. Equilibria, fixed points, and complexity classes. In *International Symp. on Theoretical Aspects of Computer Science (STACS)*, pp. 19–38, 2008.

APPENDIX

Proof of Lemma 1

PROOF. It is well known that for any polynomial computable function $f : \{0,1\}^k \mapsto \{0,1\}^m$ one can construct a circuit C with polynomial many gates that implements this function [25, Theorem 9.30]. Clearly, we can also restrict C to NOR gates with fan-in and fan-out at most 2. Organize the gates in levels according to their distance to C's output; output gates are at level 1.

We replace each gate g_i at level ℓ with the gadget in Figure 4. Nodes a, b are inputs and e is the output of the gate.

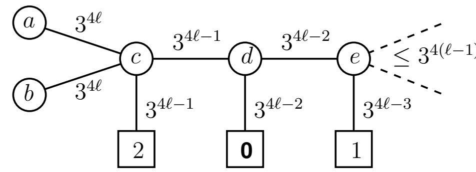


Figure 4: NOR gate

Nodes a, b and e are restricted (by supernodes) to $\{0,1\}$, node c is restricted to $\{1,2\}$, and node d is restricted to $\{0,2\}$. If a (or b) is an input of the circuit then we connect a to the corresponding input s-node by an edge of weight $3^{4\ell+1}$. If $\ell = 1$, i.e. g_i is an output gate, then we connect e to the corresponding output t-node with an edge of weight 1. Otherwise ($\ell > 1$), d is also the input to at most 2 lower level gates. The corresponding edges have weight at most $3^{4(\ell-1)}$. In any Nash-stable solution, $e = 1$ if and only if $a = b = 0$. In other words $e = NOR(a,b)$. The claim follows since our construction fulfils properties (i), (ii) and (iii). If a component of the function output has to be 2 or 3 we slightly adjust the corresponding output NOR gate. □

Proof of Lemma 2

PROOF. Since neither a nor b can improve, a' and b' are biased to $\{0,1\}$ and c is biased to 2. If $c = 2$ then the bias on a' assures $a' = \neg a$. So b' has an edge of weight w to both 0 and 1. Together with the bias this implies $b' = \neg b$. If $c = 3$ then the bias on b' assures $b' = \neg b$. So a' has an edge of weight w to both 0 and 1. Together with the bias this implies $a' = \neg a$. So in both cases $a' = \neg a$ and $b' = \neg b$. The claim follows. □

Proof of Lemma 3

PROOF. We consider three cases: (i) only a can improve, (ii) only b can improve, (iii) a and b can improve.

Case (i) (only a): Here c is biased to 3, a' is biased to 2, and b' is biased to $\{0,1\}$. First assume $c = 2$. This enforces $a' = \neg a$ which together with the bias implies $b' = \neg b$. But then the bias on c gives $c = 3$, a contradiction. Thus $c = 3$, which enforces $b' = \neg b$ and with the bias implies $a' = 2$. So a is not locked and b is locked.

Case (ii) (only b): Here c is biased to 2, a' is biased to $\{0,1\}$, and b' is biased to 3. First assume $c = 3$. This enforces $b' = \neg b$ which together with the bias implies $a' = \neg a$. But then the bias on c gives $c = 2$, a contradiction. Thus $c = 2$, which enforces $a' = \neg a$ and with the bias implies $b' = 3$. So a is locked and b is not locked.

Case (iii) (a and b): Here c is biased to 3, a' is biased to 2, and b' is biased to 3. If $c = 2$ then this enforces $a' = \neg a$, which together with the bias implies $b' = 3$. So in this case a is locked and b is not locked. If $c = 3$ then this enforces $b' = \neg b$, which together with the bias implies $a' = 2$. So in this case a is not locked and b is locked.

In every case both claims of the lemma are fulfilled. □