

A Framework for Aggregating Influenced CP-Nets and Its Resistance to Bribery

A. Maran¹, N. Maudet², Maria Silvia Pini³,
Francesca Rossi,¹ Kristen Brent Venable⁴

¹: Department of Mathematics, University of Padova, Italy
e-mail: amaran@studenti.math.unipd.it, frossi@math.unipd.it

²: LIP6, UPMC, Paris, France
e-mail: nicolas.maudet@lip6.fr

³: Department of Information Engineering, University of Padova, Italy
e-mail: pini@dei.unipd.it

⁴: Department of Computer Science, Tulane University and IHMC, USA
e-mail: kvenabl@tulane.edu

Abstract

We consider multi-agent settings where a set of agents want to take a collective decision, based on their preferences over the possible candidate options. While agents have their initial inclination, they may interact and influence each other, and therefore modify their preferences, until hopefully they reach a stable state and declare their final inclination. At that point, a voting rule is used to aggregate the agents' preferences and generate the collective decision. Recent work has modeled the influence phenomenon in the case of voting over a single issue. Here we generalize this model to account for preferences over combinatorially structured domains including several issues. We propose a way to model influence when agents express their preferences as CP-nets. We define two procedures for aggregating preferences in this scenario, by interleaving voting and influence convergence, and study their resistance to bribery.

People often exchange opinions before taking a decision (Krackhardt 1987; Grabisch and Rusinowska 2012). In assemblies, "rules of order" typically prescribe a debate to take place before actual voting. When managers discuss whether to introduce new technologies in a company, the discussion may take many rounds, in which the initial opinions may change because of the influence of what others say, until a stable set of opinions is formed; at that point, voting can take place and a collective decision is taken. Polls in political elections provide a representative sample of the opinions of the voters, and may induce some voters to change their mind about the candidates. Also, in a marketing context involving complex choices, some people may be identified as followers or prescribers for certain features. Moreover, influences can be used to model hierarchical organizations.

We consider scenarios where a set of agents need to take a collective decision by voting over the possible candidate decisions, and may exchange information before actually declaring their final vote. We assume that the information agents exchange is the mere observation of others' vote:

agents may revise their vote on the basis of the observed votes of others. In other words, agents may *influence* each other, leading to their preferences be modified accordingly. Influence is usually an iterative process, during which agents can be at the same time influencing and influenced entities, so they may change their inclination more than once based on the changes in the preferences of other agents. Some influence schemes may converge, while others may loop. The concept of influence has been widely studied in psychology, economics, sociology, and mathematics (DeGroot 1974; DeMarzo and Vayanos 2003; Krause 2000). An overview of dynamic models of social influence can be found in (Jackson 2008). Recent work has formally modelled and studied the influence phenomenon in the case of taking a decision over a single two-values issue (Grabisch and Rusinowska 2011). In the context of human users these influence schemes arise from well-studied social models that are estimated via polls, while in the context of artificial agents, the influences can represent natural hierarchical organizations of agents.

Here we generalize these schemes and models to account for preferences over combinatorially structured domains including several issues that may be dependent on each other. In fact, the set of possible decisions, over which agents express their preferences, may have a combinatorial structure, that is, each candidate decision can be seen as the combination of certain issues, where each issue has a set of possible instances. Even if there are few issues and instances, they could give rise to a large number of candidate decisions. A compact way to express one's preferences over such a large set is preferable, otherwise too much space would be needed to rank all possible alternatives. CP-nets are a successful framework that allows one to do this (Boutilier et al. 2004). They exploit the independence among some issues to give conditional preferences over small subsets of them. CP-nets have already been considered in a multi-agent voting setting (Rossi, Venable, and Walsh 2004; Lang and Xia 2009; Purrington and Durfee 2007; Xia, Conitzer, and Lang 2008; Mattei et al. 2013). Here we adapt such frameworks to incorporate influences among agents, by allowing influences to be over the same issue or also among different issues. An

interesting feature of our model is that influence is embedded smoothly in the multi-agent CP-net profile, and there is a convenient coincidence between the optimal outcomes of certain CP-nets and the stable states of the influence iterative process.

To aggregate preferences in this framework, we propose two procedures, called Finally Aggregation (FA) and Level Aggregation (LA), to find a collective decision by interleaving voting and influence convergence. FA performs influence iteration at each level of the CP-nets and it aggregates agents' preferences only at the end, while LA performs influence iteration and preference aggregation at each level. We then evaluate such procedures in terms of resistance to bribery. Bribery in voting may be regarded as a type of influence, although it does not involve an iterative process: an external agent (the briber) wants to influence the result of the voting process by convincing some agents to change their vote, in order to get a collective result which is more preferred to him; there is usually a limited budget to be spent by the briber to convince agents (Faliszewski, Hemaspaandra, and Hemaspaandra 2009). We show that the presence of inter-agent influence can make bribery computationally difficult, even in a very restrictive setting, both for LA and FA. On the other hand, there are cases where bribery can be computationally easy for LA. The paper is a revised and extended version of (Maudet et al. 2012).

Background

Influence Functions

In (Hoede and Bakker 1982; Grabisch and Rusinowska 2010a; 2010b; 2011) a framework to model influences among agents in a social network environment is defined. Each agent has two possible actions to take and it has an inclination to choose one of the actions. Due to influence by other agents, the decision of the agent may be different from its original inclination. The transformation from the agent's inclination to its decision is represented by an influence function. In many real scenarios, influence among agents does not stop after one step but it is an iterative process. Formally, an *influence function* B over n agents is a function that maps every vector of inclinations $I = (I_1, \dots, I_n) \in \{-1, +1\}^n$, where I_i is the inclination of the agent i , into a vector of decisions $B(I) = (B_1(I), \dots, B_n(I)) \in \{-1, +1\}^n$, where $B_i(I)$ denotes the decision made by the agent i . *Stable states* satisfy $I_i^{(k)} = I_i^{(k+1)}$, for every agent i , starting from a certain k , where k is the number of iterations and $I_i^{(k)}$ denotes the inclination (*state*) of agent i at the iteration k . A set of agents such that their $B_i(I)$ coincide in a stable state is a consensus group. The influence function can be modelled via a graph where nodes are states and arcs model state transitions via the influence function. Starting from an initial state, via the influence function we may pass from state to state until stability holds (in the graph formulation, we are in a state represented by a node with a loop), or we may also not converge. Here are some examples from (Grabisch and Rusinowska 2011):

Fol is an influence function between two agents, each following the inclination of the other one. It converges to stability when the initial inclination is a consensus between the two agents. Otherwise, influence iteration never stops.

Gur is an influence function where one of the agents is the guru and all other agents follow him. It has two stable states, which both represent consensus. Given any initial inclination, the iteration will converge to one of the stable states.

Conf3 models a community with a king, a man, a woman, and a child, following a Confucian model: the man follows the king, the woman and child follow the man, and the king is influenced by others only if he has a positive inclination, in which case he will follow such an inclination only if at least one of the other people agrees with him. This influence function always converges to one of two stable states, which both represent consensus, depending on the initial state.

CP-nets

CP-nets (Boutilier et al. 2004) (for Conditional Preference networks) are a graphical model for compactly representing conditional and qualitative preference relations. They are sets of *ceteris paribus* preference statements (cp-statements). For instance, the cp-statement "*I prefer red wine to white wine if meat is served.*" asserts that, given two meals that differ *only* in the kind of wine served *and* both containing meat, the meal with red wine is preferable to the meal with white wine. Formally, a CP-net has a set of features $F = \{x_1, \dots, x_n\}$ with finite domains $\mathcal{D}(x_1), \dots, \mathcal{D}(x_n)$. For each feature x_i , we are given a set of *parent* features $Pa(x_i)$ that can affect the preferences over the values of x_i . This defines a *dependency graph* in which each node x_i has $Pa(x_i)$ as its immediate predecessors. An *acyclic* CP-net is one in which the dependency graph is acyclic. Given this structural information, one needs to specify the preference over the values of each variable x for *each complete assignment* on $Pa(x)$. This preference is assumed to take the form of a total or partial order over $\mathcal{D}(x)$. A cp-statement has the general form $x_1 = v_1, \dots, x_n = v_n : x = a_1 \succ \dots \succ x = a_m$, where $Pa(x) = \{x_1, \dots, x_n\}$, $\mathcal{D}(x) = \{a_1, \dots, a_m\}$, and \succ is a total order over such a domain. The set of cp-statements regarding a certain variable X is called the cp-table for X .

Consider a CP-net whose features are A, B, C , and D , with binary domains containing f and \bar{f} if F is the name of the feature, and with the cp-statements as follows: $a \succ \bar{a}$, $b \succ \bar{b}$, $(a \wedge b) : c \succ \bar{c}$, $(\bar{a} \wedge \bar{b}) : c \succ \bar{c}$, $(a \wedge \bar{b}) : \bar{c} \succ c$, $(\bar{a} \wedge b) : \bar{c} \succ c$, $c : d \succ \bar{d}$, $\bar{c} : \bar{d} \succ d$. Here, statement $a \succ \bar{a}$ represents the unconditional preference for $A = a$ over $A = \bar{a}$, while statement $c : d \succ \bar{d}$ states that $D = d$ is preferred to $D = \bar{d}$, given that $C = c$.

A *worsening flip* is a change in the value of a variable to a less preferred value according to the cp-statement for that variable. For example, in the CP-net above, passing from $abcd$ to $ab\bar{c}d$ is a worsening flip since c is better than \bar{c} given a and b . One outcome α is *better* than another outcome β (written $\alpha \succ \beta$) iff there is a chain of worsening flips from α to β . This definition induces a preorder over the outcomes,

which is a partial order if the CP-net is acyclic.

Finding the optimal outcome of a CP-net is NP-hard (Boutilier et al. 2004). However, in acyclic CP-nets, there is only one optimal outcome and this can be found in linear time by sweeping through the CP-net, assigning the most preferred values in the cp-tables. For instance, in the CP-net above, we would choose $A = a$ and $B = b$, then $C = c$, and then $D = d$. In the general case, the optimal outcomes coincide with the solutions of a set of constraints obtained replacing each cp-statement with a constraint (Brafman and Dimopoulos 2004): from the cp-statement $x_1 = v_1, \dots, x_n = v_n : x = a_1 \succ \dots \succ x = a_m$ we get the constraint $v_1, \dots, v_n \Rightarrow a_1$. For example, the following cp-statement (of the example above) $(a \wedge b) : c \succ \bar{c}$ would be replaced by the constraint $(a \wedge b) \Rightarrow c$.

Modeling Influence within Profiles

In our setting n agents express their preferences over a set of candidates with a combinatorial structure: there are m features and each candidate is an assignment of values to all features. We assume features to be binary (that is, with two values in their domain). Agents' preferences over the candidates are modeled via acyclic CP-nets. Moreover, the dependency graphs of such CP-nets must all be compatible with a linear order O over the features: for each voter, the preference over a feature is independent of features following it in O ¹. This implies that the n CP-nets N_1, \dots, N_n are such that the union of their dependency graphs, that we call $Dep(N_1, \dots, N_n)$, does not contain cycles. Notice that CP-nets with this property may have different dependency graphs.

Definition 1 (profile) *Given n agents, m binary features, and a linear ordering O over the features, a profile is a collection of n acyclic CP-nets over the m features which are compatible with O .*

A profile models the initial inclination of all agents, that is, their opinions over the candidates before they are influenced by each other. Since the set of features is the same for all agents, but each agent may have a possibly different CP-net, to avoid confusion we call variables the binary entities of each CP-net. Thus, given a profile with m features, for each feature there are n variables modelling it, one for each CP-net. Thus the whole profile has $m * n$ variables. Given a profile P with CP-nets N_1, \dots, N_n , we will often write $Dep(P)$ to mean $Dep(N_1, \dots, N_n)$.

In (Grabisch and Rusinowska 2011) influence functions act on each single feature: the preferences of an agent over a certain feature may be influenced by the preferences of one or more other agents over the same feature. Also, only positive influence is allowed. We adopt a more general notion of influence, that could be either positive or negative, and it could also be across features. For example, an agent could say "if Alice doesn't prefer pasta, I would like to take pasta". Or also, "if Bob prefers to go out tomorrow, I prefer to go for dinner", or even "if Alice prefers to drink wine, I will follow

her preferences, otherwise I will follow my inclination". We do not allow for conflicting influence statements. We believe this is reasonable as it is comparable to not allowing irrational preferences. Each influence function is modeled via one or more conditional influence statements.

Definition 2 (ci-statement and ci-table) *A conditional influence statement (ci-statement) on variable X has the form $o(X_1), \dots, o(X_k) :: o(X)$, where $o(Y)$ is an ordering over the values of variable Y , for $Y \in \{X_1, \dots, X_n, X\}$. Variables X_1, \dots, X_k are the influencing variables and variable X is the influenced variable. A ci-table is a collection of ci-statements with the same influenced variable, and such that any pair of influencing contexts is mutually exclusive.*

A ci-statement models the influence on variable X of the preferences over a set of influencing variables X_1, \dots, X_k . Such preferences are given by the agents owning the variables. Note that the semantics differs from the one used in cp-statements. Such a ci-statement must be interpreted as asserting the preferences over X in the specified influencing context, *disregarding the other variables*. Also, unlike a cp-table, a ci-table may not specify the values of the influenced variable for all possible assignments of the influencing variables. Since we are dealing with binary variables, we will compactly specify an ordering over the values of a variable by writing just the top element.

An agent *positively influences* (resp. *negatively influences*) another agent when there are (resp. there are no) circumstances under which this other agent will adopt the same top ranked option as the influencing agent, and no (resp. there are) circumstances where he would deliberately pick a different one. If i always adopts the same top as j , we say that i follows j .

We are now ready to model influence functions, such as those defined in (Grabisch and Rusinowska 2011). For the **Conf3** influence function, we recall that the king is influenced by others only if he has a positive inclination, in which case he will follow such an inclination only if at least one of the other people agrees with him. We may use a single binary feature X and 4 binary variables $X_k, X_m, X_w,$ and X_c . Each variable X_i , with $i \in \{k, m, w, c\}$, has two values denoted by x_i and \bar{x}_i . The ci-statement modelling the influences over the king are: $\bar{x}_k :: \bar{x}_k, x_k \bar{x}_m \bar{x}_w \bar{x}_c :: \bar{x}_k, x_k x_m :: x_k, x_k x_w :: x_k,$ and $x_k x_c :: x_k$. Even if in this example we have a small number of ci-statements, a general mapping from any influence function to a set of ci-statements, will produce between 1 and $n \times 2^n$ ci-statements if we have n agents. Given an influence function f , we will call $ci(f)$ the ci-statements modelling f .

A Characterization of Stable States

We may notice that ci- and cp-statements are similar in syntax. They are also linked in terms of their semantics: when the ci-statements $ci(f)$ of an influence function f are interpreted as cp-statements and turned into corresponding constraints, the optimal outcomes corresponding to the set of such cp-statements coincide with the stable states of f .

Theorem 1 *Given an influence function f , consider the set of cp-statements S corresponding to the ci-statements $ci(f)$.*

¹This coincides with the notion of O -legality in (Lang and Xia 2009).

The undominated outcomes of S coincide with the stable states of f .

Proof: Since we may not have a ci-statement for every possible ordering over the influencing variables, S may not be a CP-net. Nevertheless, the cp-statements induce an ordering over the outcomes. Let $o = (X_1 = x_1, \dots, X_t = x_t)$ be an undominated outcome in this ordering. This means that flipping any value in o is never improving in any cp-statement. Thus there cannot be any ci-statement for which the assignment in o of the influenced variable would be changed given the assignments in o to its influencing variables. Thus o is a stable state for the influence function. Now let d be a dominated outcome according to the ordering induced by S . If it is dominated, it must be dominated by at least one outcome, say d' , that is one flip away. Let X be the variable on which they differ. Since d' dominates d , there must be a cp-statement on X where the parents of X are assigned the values they have in d (and d'), and according to which the value X in d' is preferred to the one in d . Applying the influence function to the state corresponding d would induce a change in the inclination on X . This allows us to conclude that this state is not stable. \square

In the example above, if we interpret the ci-statements as cp-statements and write the corresponding constraints, we get: for the king: $(\bar{x}_k \Rightarrow \bar{x}_k), (x_k \bar{x}_m \bar{x}_w \bar{x}_c \Rightarrow \bar{x}_k), (x_k x_m \Rightarrow x_k), (x_k x_w \Rightarrow x_k), (x_k x_c \Rightarrow x_k)$; for the man: $(x_k \Rightarrow x_m), (\bar{x}_k \Rightarrow \bar{x}_m)$; for the woman: $(x_m \Rightarrow x_w), (\bar{x}_m \Rightarrow \bar{x}_w)$; for the child: $(x_m \Rightarrow x_c), (\bar{x}_m \Rightarrow \bar{x}_c)$. The only two solutions of this set of constraints are the assignments (x_k, x_m, x_w, x_c) and $(\bar{x}_k, \bar{x}_m, \bar{x}_w, \bar{x}_c)$, which are exactly the two stable states of the **Conf3** influence function.

Theorem 1 may suggest that ci-tables may be turned into cp-tables in the profile, thus getting rid of ci-statements. However, finding all the stable states of a function is not sufficient: we need to know if a stable state can be reached from a given initial inclination, or if no stable state can be reached. We need to specify the dynamics of influence.

Influence Iteration

We adopt the following approach for every feature: (1) agents declare their initial inclination regarding the feature; (2) agents consider their ci-tables and then simultaneously declare whether they stick to their opinion or change it (by influence); (3) if all agents stick to their opinion, the state is stable, otherwise the process is iterated.

To find a stable state, or to find out that there is no stable state reachable from the initial inclination, we apply these steps iteratively. We first consider all variables regarding the same feature, and start with the assignment s of such variables modeling the initial inclination. We then move to another assignment s' by setting the value of each variable with an ingoing influence link to its most preferred value, given the values in s of its influencing variables. We then iterate this step until we either (i) reach a state which has been seen one step before (that is, a stable state), or (ii) reach a state already seen at least two steps before (that is, a cycle is detected). In that case, a policy has to return a single state (here we assume to select one of the states in the cycle).

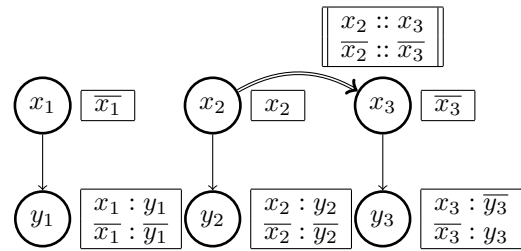
Influenced CP-nets

In line with the CP-nets graphical notation, we use hyperarcs to graphically model influences. They go from the influencing variables to the influenced variable. To distinguish them from the dependencies, we call them ci-arcs. Notice that we consider acyclic CP-nets, while ci-arcs may create loops due to the iterative nature of influences: a self-influencing variable models the fact that the value of the variable in the next state depends on its value in the current state.

Definition 3 (I-profile) An I-profile is a triple (P, O, S) , where P is a profile composed by n CP-nets compatible with O , an ordering over the m features of P , and S is a set of ci-tables.

As we said above, O must be such that $Dep(P)$ has only arcs from earlier variables to later variables. This ordering partitions the set of variables into m levels. Variables in the same level correspond to the same feature. Moreover, we assume that each variable can be influenced only by variables in her level or in earlier levels, but not both, in the same ci-statement. Because of this restriction, ci-arcs in an I-profile can create cycles only among variables of the same level. Notice that variables may appear both in ci-tables and cp-tables: influences and conditional preferences may be in conflict. In this case, *influences override preferences*.

Consider the I-profile below. There are three agents and thus three CP-nets with two binary features: X and Y . The ordering O is $X \succ Y$. Each variable X_i (resp., Y_i), with $i \in \{1, 2, 3\}$, has two values denoted by x_i and \bar{x}_i (resp., y_i and \bar{y}_i). Value x_i for the variables X_i correspond to value x for X , and similarly for Y . Variables X_i belong to the first level while variables Y_i belong to the second level. cp-statements are denoted by single-line arrows while ci-statements are denoted by doubled-line arrows. Agent 3 is influenced (positively) on feature X by agent 2.



Aggregating Influenced Preferences

To aggregate agents' preferences contained in an I-profile, while taking into account the influence functions, we define two procedures based on a sequential approach similar to the one considered in (Lang and Xia 2009), where at each step we consider one of the features, in the ordering stated by the I-profile. Both procedures include three main phases: (i) *influence iteration* within one level, (ii) *propagation* from one level to the next one, and (iii) *preference aggregation*. In both cases, at the end, a winner candidate will be selected, that is, a value for each feature. Notice that, for each feature, we consider the influences among different variables modelling this feature. At the first level, the variables are

all independent in terms of cp-statements, so each agent has an initial inclination over the values of his variable which does not depend on any other variable. For the other levels, the initial inclination is obtained by propagating information from the previous levels.

Level Aggregation (LA): At each level (starting from the first one, where there are only independent variables), we perform first *influence iteration* over every feature of this level and then we *aggregate* influenced preferences to obtain a collective value for this feature. Since variables are binary, we aggregate preferences over the variables by using the Majority rule. Ties are broken with a tie-breaking rule where precedence is given by a lexicographical ordering where the features are ordered as O and for every feature X , $\bar{x} \succ x$. Then, we *propagate* the selected value for the feature to the next level. More precisely, for variables with incoming ci-arcs from previous levels, we set their initial inclination according to these ci-statements; for variables with no incoming ci-statements, their initial inclination is determined by their cp-table according to the collective value chosen for the variables of the previous levels.

Final Aggregation (FA): At each level (starting from the highest one), we perform *influence iteration* and thus each variable in the considered level has a final inclination, that is, an ordering for its values. Then, we *propagate* this information to the next level. For variables with incoming ci-arcs from previous levels, we set the initial inclination according to these ci-statements and the final ordering of the influencing variables (just like for LA). Instead, for variables with no incoming ci-statements, their initial inclination is determined by their cp-table according to the top element of the ordering of the variables of the previous levels. After all levels have been processed this way, for every CP-net we have an assignment of values for every feature. At that point we perform *preference aggregation* of these outcomes via the Plurality rule, which returns the outcome which is the most preferred by the greatest number of agents. Ties are broken with the same tie-breaking rule as for LA.

A simple sufficient condition for LA and FA to return the same winner is that there exists a set of agents which constitute a consensus group of size at least $n/2$ on all features. This would be the case in our example if agent 2 followed agent 3 on feature Y . But in general LA and FA may yield different results. In the I-profile of our example, after the influence iteration step at level 1 (that is, on feature X), the preference of agent 3 is $x_3 \succ \bar{x}_3$, while the preferences of the other agents are unchanged. If we use LA, we aggregate the votes over X by majority. This results in $X = x$ winning and thus the variables of the first level are set to the values: $X_1 = x_1$, $X_2 = x_2$, and $X_3 = x_3$. We then propagate such assignments to the next level and we get the following assignment for the variables corresponding to the Y feature: $Y_1 = y_1$, $Y_2 = y_2$, and $Y_3 = \bar{y}_3$. We now aggregate the votes over Y by majority, and the winning assignment is $Y = y$. The overall winner of the procedure is $(X = x, Y = y)$. Instead, with FA, the assignments for X that are propagated are those after the influence iteration, i.e., $X_1 = \bar{x}_1$, $X_2 = x_2$, and $X_3 = x_3$. This gives, through propagation, the following values for the variables corre-

sponding to Y : $Y_1 = \bar{y}_1$, $Y_2 = y_2$, and $Y_3 = \bar{y}_3$. Thus we have the following top candidates: $C1 = (X = \bar{x}, Y = \bar{y})$, $C2 = (X = x, Y = y)$, and $C3 = (X = x, Y = \bar{y})$ and the winner is $C1 = (X = \bar{x}, Y = \bar{y})$.

The choice of the ordering O does not influence the winner, no matter if we use LA or FA.

Theorem 2 *Given two I-profile (P, O, S) and (P, O', S) , their winners coincide, if we use the same aggregation method (either LA or FA).*

Proof: Different orderings of an I-profile with the same profile and the same ci-statements will possibly order differently only variables that are independent both in terms of preferences and influence functions. \square

LA and FA differ only in whether the result of the vote on each variable is propagated or not. There are scenarios where such a result can be broadcasted to the voters, as well as scenarios where security or privacy concerns may lead to prefer minimal communication between the system aggregating the preferences and the voters.

Bribery in LA and FA

In the traditional voting setting, a briber is an outside agent with a limited budget that attempts to affect the outcome of an election by paying some of the agents to change their preferences (Faliszewski, Hemaspaandra, and Hemaspaandra 2009). Our scenario is particularly interesting since there are influential agents which, once bribed, can lead to a change in the inclination of other agents at no additional cost. In our setting, we can conceive a model where bribery takes place *before* the actual dynamics of influence occurs (that is, only initial inclination are modified), or is interleaved with it (that is, at each step of the influence interaction, the briber can pay some agents to change their preferences). In this paper we assume bribery occurs before the dynamics of influence. An agent i may charge a cost $c_{i,X}$ for each feature X for which he is asked to change his inclination. Influence and bribery can result in contradicting pressures: it may be the case that (i) *influence overrides bribery*, noted $i \gg b$, or that (ii) *bribery overrides influence*, noted $b \gg i$, in which case, bribing means fixing the value of the feature for good by discarding influences potentially affecting this value. Both schemes can be naturally integrated in our model, by considering a cost required to override influence, and a smaller cost which may suffice to affect the initial inclination only. Intuitively, the presence of influences may favor the briber by making bribery cheaper. However, from a computational point of view, influences make the problem difficult for him, even in a very restrictive setting.

Theorem 3 *Let P be an I-profile with positive acyclic influences, where each agent i charges a bribery cost $c_{i,X}$ for each feature X . Given a candidate p and a budget k , deciding whether p can be made the winner within k is NP-complete both for LA and FA, under both the $b \gg i$ and the $i \gg b$ scheme. This holds even if the cost is the same for all agents and all features.*

Proof: It is in NP, since, given a set of agents to bribe, it is possible to test in polynomial time if the winner is p and

if the budget does not exceed k . To show NP-completeness, we use a reduction from X3C. In X3C we are given an instance (B, S) of X3C, where $B = \{b_1, \dots, b_{3k}\}$ is some set and $S = S_1, \dots, S_n$ is a family of 3-element subsets of B . We ask if there is a collection of exactly k sets in S so that their union is B . We create a bribery instance on an I-profile with a single binary feature X with values x and \bar{x} , and $n+3k+(n+3k-1)$ voters. The first n voters correspond to the sets in S , the next $3k$ voters correspond to elements b_1, \dots, b_{3k} , and the last $(n+3k-1)$ voters are dummies. Voters in $S \cup B$ and $4k-1$ dummy voters vote \bar{x} . The other voters vote x . Thus, we have $n+3k+4k-1 = n+7k-1$ votes for \bar{x} and $2n+6k-1-(n+7k-1) = n-k$ votes for x . Further, the costs are set for all i as $c_{i,X} = 1$. The budget is set to k . The (positive) influences are defined as follows: an agent b_i is influenced by agent S_j if and only if b_i belongs to S_j . It is easy to see that it is possible to ensure that x wins by bribery, only if there is a way to pick k sets from S such that their union is B . If such sets exist, then after the bribery we have additional $4k$ votes for x (altogether there are $n+3k$ votes for x , and x is chosen). On the other hand, if there is no way to pick such k sets, then by bribing k voters we cannot get more than $4k-1$ voters to switch from \bar{x} to x , so bribery is impossible. Finally, as a YES instance requires to spend all the budget on non influenced voters (from S), the choice of $i \gg b$ or $b \gg i$ is irrelevant, and the hardness result holds for both schemes. \square

This result does not rely much on the voting rule used with FA: hardness holds also for rules that coincide with simple majority with 2 candidates.

Assuming the cost is the same for all the features at the same level (since the importance of these features is the same in every CP-net), we have identified influence functions for which bribery is easy for LA with $i \gg b$.

Theorem 4 *Let P be an I-profile where influences are of type **Gur**, **Fol**, **Conf3**, or they are acyclic positive or negative influences where each agent is influenced by at most one other agent. Assume that the cost of bribing an agent on a feature at a given level is the same for all agents. Then, given a candidate p and a budget k , deciding whether p can be made the winner within k is in P for LA.*

Proof: We transform the bribery problem at each level into a problem of bribery with weighted agents and a fixed set of costs. The overall algorithm applies, to each level, first bribery, then influence, and then aggregation. After each step at each level, intra-level propagation takes place. In what follows we assume the briber is pushing for value 1. The key step is the computation of the minimal bribery cost at each level. If the sum of the costs on all levels does not exceed the budget, we accept, otherwise we reject. For each level, we first compute the bribery cost for each cluster of voters. The cluster size is 2 in the case of function **Fol**, 4 for **Conf3** and can be anywhere between 2 and n for **Gur**. We assume that each agent belongs to a unique cluster. It is possible to compute the minimum cost for bribing a cluster subject to one of the above influence functions in the linear time in the size of the cluster. For **Fol**, either both agents agree with each other and with the briber (cost 0), or they

both disagree with the briber (cost 2) or only one of them disagrees and needs to be bribed (cost 1); for **Gur**, if the guru agrees we have cost 0, otherwise cost 1; for **Conf3**, if the king and one other agent of the cluster agree with the briber we have cost 0; if at least one agent, other than the king, agrees with the briber then we have cost 1 for bribing the king; otherwise, cost 2 for bribing the king and another agent. We then set the vote on all of those clusters with cost 0 to be in favor of the briber (i.e., equal to the projection of p on the current feature). If we have not reached the majority, we replace each **Fol** cluster with a single agent of weight 2 and cost either 1 or 2, depending on how many agents must be bribed. We replace every **Conf3** cluster with a single agent with weight 4 and cost either 1 or 2, and every **Gur** cluster with a single agent with weight equal to the number of agents in the cluster and cost 1. For each agent, not in any cluster and not influenced by other agents, we compute the number of votes he brings in favor of the briber, both when he votes for and against p (this is due to the presence of negative influences). Let t be the maximum gain that such an agent can bring. We replace the agent and all the agents influenced (directly or indirectly) by him with a single agent with cost 1 and weight t . We then order all the new agents with weight greater than or equal to 3 and cost 1 in decreasing order of weight and we bribe them following such an ordering. If by doing so we don't exceed the budget or we do not obtain a majority, we continue by bribing the agents with weight 4 and cost 2, then the agents with cost 1 and weight 2, then the agents with cost 2 and weight 2, and finally the ones with cost 1 and weight 1. This until we either exceed the budget or we obtain a majority. The only caution that should be taken is that if the budget is exceeded by bribing a voter of cost 2, then the following ones with cost 1 should be considered. Assuming the budget is not exceeded, influences are applied and the result is propagated to the next level where the procedure is iterated given the residual budget. \square

Conclusions and Future Work

We studied a framework for handling influence in the context of the aggregation of agents' preferences expressed via CP-nets. The paper bridges the gap between influences in a single-issue setting and the CP-net approach to model preferences over combinatorial domains, leading to original bribery issues. Bribery in voting with CP-nets has been considered also in (Mattei et al. 2012); however, scenarios with influences among agents had not been investigated before.

When a single feature is considered in isolation and variables are binary, our model is similar to Boolean networks (BN). Such networks are studied in biology as a model of genetic networks, and specify how nodes vary depending on the input nodes, as specified by a regulation function. In the synchronous model, nodes change their values all at the same time. This corresponds to our influence dynamics. We plan to exploit BN's algorithms to search for stable states and to study the connection between our bribery problem and the problem of controlling genetic networks (Akutsu et al. 2007). Recently the pressure of peers over the preferences of agents has been considered (Liang and Seligman 2011), distinguishing different strength of suggestion in a

logical setting, and studying the dynamics of such suggestions, given some model of preference change. We intend also to analyze the computational complexity of manipulating LA and FA, to study other normative properties of such procedures, as well as to include probabilistic influence schemes and influences the ordering among a set of possible actions, as in (Grabisch and Rusinowska 2010b).

References

- Akutsu, T.; Hayashida, M.; Ching, W.-K.; and Ng, M. K. 2007. Control of boolean networks: hardness results and algorithms for tree-structured networks. *Journal of Theoretical Biology* 244:670–679.
- Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *JAIR* 21:135–191.
- Brafman, R., and Dimopoulos, Y. 2004. Extended semantics and optimization algorithms for cp-networks. *Computational Intelligence* 20(2):218–245.
- DeGroot, M. 1974. Reaching a consensus. *Journal of the American Statistical Association* 69:118–121.
- DeMarzo, P., and Vayanos, D. 2003. Persuasion bias, social influence, and unidimensional opinions. *Quarterly Journal of Economics* 118:909–968.
- Faliszewski, P.; Hemaspaandra, E.; and Hemaspaandra, L. A. 2009. How hard is bribery in elections? *JAIR* 35:485–532.
- Grabisch, M., and Rusinowska, A. 2010a. A model of influence in a social network. *Theory and Decisions* 69(1):69–96.
- Grabisch, M., and Rusinowska, A. 2010b. A model of influence with an ordered set of possible actions. *Theory and Decisions* 69(4):635–656.
- Grabisch, M., and Rusinowska, A. 2011. Iterating influence between players in a social network. In *Proc. 16th Coalition Theory Network Workshop*.
- Grabisch, M., and Rusinowska, A. 2012. A model of influence-based on aggregation functions. *Working paper*.
- Hoede, C., and Bakker, R. 1982. A theory of decisional power. *Journal of Mathematical Sociology* 8:309–322.
- Jackson, M. O. 2008. *Social and Economic Networks*. Princeton University Press.
- Krackhardt, D. 1987. Cognitive social structures. *Social Networks* 9:109–134.
- Krause, U. 2000. A discrete nonlinear and nonautonomous model of consensus formation. *Communications in Difference Equations*.
- Lang, J., and Xia, L. 2009. Sequential composition of voting rules in multi-issue domains. *Mathematical social sciences* 57:304–324.
- Liang, Z., and Seligman, J. 2011. The dynamics of peer pressure. In *Proceedings of the Third international conference on Logic, rationality, and interaction, LORI'11*, 390–391.
- Mattei, N.; Rossi, F.; Venable, K. B.; and Pini, M. S. 2012. Bribery in voting over combinatorial domains is easy. In *Proc. ISAIM 2012*.
- Mattei, N.; Pini, M. S.; Venable, K. B.; and Rossi, F. 2013. Bribery in voting with CP-nets. *Annals of Mathematics and Artificial Intelligence*.
- Maudet, N.; Pini, M. S.; Rossi, F.; and Venable, K. B. 2012. Influence and aggregation of preferences over combinatorial domains. In *Proc. AAMAS 2012*, 1313–1314.
- Purrington, K., and Durfee, E. H. 2007. Making social choices from individuals' cp-nets. In *AAMAS*, 179.
- Rossi, F.; Venable, K.; and Walsh, T. 2004. mCP nets: Representing and reasoning with preferences of multiple agents. In *AAAI-04*, 729–734.
- Xia, L.; Conitzer, V.; and Lang, J. 2008. Voting on multi-attribute domains with cyclic preferential dependencies. In *AAAI*, 202–207.