Bribery in Voting Over Combinatorial Domains Is Easy

(Extended Abstract)

Nicholas Mattei
University of Kentucky
Lexington, KY, USA
nick.mattei@uky.edu

Maria Silvia Pini
University of Padova
Padova, Italy
mpini@math.unipd.it

Francesca Rossi
University of Padova
Padova, Italy
frossi@math.unipd.it

K. Brent Venable
University of Padova
Padova, Italy
kvenable@math.unipd.it

ABSTRACT

We investigate the computational complexity of finding optimal bribery schemes in voting domains where the candidate set is the Cartesian product of a set of variables and agents' preferences are represented as CP-nets. We show that, in most cases, the bribery problem is easy. This also holds for some cases of k-approval, where bribery is difficult in traditional domains.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*; F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Theory, Algorithms

Keywords

Voting Protocols, Bribery, Manipulation, Complexity

1. INTRODUCTION

Making collective decisions is a challenging task for both humans and autonomous agents. Computational social choice focuses on computational questions regarding group decision making [3]. In this document we consider a scenario where a collection of agents use the CP-net formalism to compactly represent their preferences over a common set of issues that may have conditional dependencies [2].

When voting [1] is structured as the combination of several decisions, one natural method to determine a winner is to decide on an issue by issue basis, while the other natural approach is to aggregate the agents' votes over complete combinations of issues. We consider both approaches and we study elections via sequential (that is, issue by issue) majority (SM), plurality (OP), veto (OV), and k-approval (OK).

In this setting, we study the bribery problem, which is when an outside agent with a limited budget attempts to affect the outcome of an election by paying some of the agents to change their preferences [6]. We consider several cost schemes to compute the cost of changing a vote of an agent in response to a briber's request, which are based on the actual changes to be made in the CP-net. We show that, in most cases, bribery in combinatorial domains is easy. This also holds for some cases of OK, where bribery is difficult in traditional domains.

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

2. VOTING WITH CP-NETS

Our setting consists of a set of n agents with preferences over a common set of candidates with a combinatorial structure: there is a common set of m binary issues and the set of candidates is the Cartesian product of their domains. Each candidate (or outcome) is an assignment of values to all issues, thus we have 2^m candidates.

We assume each agent expresses its preferences over the candidates via an acyclic CP-net [2]. CP-nets are sets of *conditional preference statements* (cp-statements) each stating a total order over the values of a variable (say X), possibly depending on each combination of values of a set of other variables (say $X_1, \ldots, X_n$). X is said the dependent variable and $X_1, \ldots, X_n$ are the parents of X. Acyclic CP-nets are CP-nets where the dependency graph (with arcs from parents to dependent variables) does not have cycles.

Voting theory [1] provides many voting rules to aggregate agents' preferences. Each rule takes, as input, a partial or complete preference ordering of the agents and gives, as output, the "winner" outcome (the best outcome according to the rule). When there are more than two candidates, there are many voting rules one could use and we consider three. In **plurality** the candidate ranked in first place by the largest number of voters wins. When there are two candidates, plurality coincides with majority. In **veto** each voter chooses a candidate to veto and the candidate with the least number of vetoes wins. In **k-approval** each voter labels k candidates as approved or not and the candidate with the most approvals wins.

When we use a sequential approach to voting, we require a total ordering O over the issues so that, in each CP-net, each issue must be independent of all issues following it in the ordering O. A *profile* (P, O) is a collection P of n CP-nets over m common issues and a total ordering O over the issues that satisfies the above property. This is called an O-legal profile in [7]. The CP-nets appearing in such profiles do not necessarily have the same dependency graphs.

3. COMBINATORIAL BRIBERY

The bribery problem we consider is parametric with respect to three items: the way a winner is chosen from the given profile, the

	SM	SMw	OP(IV) OV(IV) OK*(IV)	OP(DV,IV+DV) OV(DV,IV+DV) OK*(DV,IV+DV)
C_{EQUAL}	NP-c	NP-c	P	P
C_{FLIP}	P	NP-c	P	P
C_{LEVEL}	P	NP-c	P	?

Table 1: Our complexity results for the combinatorial bribery problem. OK* stands for OK when k is a power of 2.

allowed bribery actions, and the cost scheme for such actions.

In our domain, agents have a CP-net instead of an explicit outcome ordering. Therefore, we define the bribery actions as changes made directly to the cp-statements within the CP-net of an agent. Since we consider binary issues, changing a cp-statement means flipping the positions of the two values of an issue. In a CP-net, a cp-statement is associated to a certain issue, and issues are of two kinds: independent and dependent. We distinguish bribery actions on these two kinds of issues denoting with IV (resp. DV) the situation in which the briber asks for flips only in cp-statement related to independent (resp. dependent) issues. When both are allowed, we write IV+DV.

Also the cost schemes we consider are defined in terms of the amount of change the agents have to make on their CP-nets in order to comply with the briber's request. In particular, we consider:

C_{EQUAL}: A unit cost allows any number of flips in a CP-net.

C_{FLIP}: The cost is the total number of flips in the CP-net.

C_{LEVEL}: An issue which is closer to be independent is regarded as more important. We may link this importance to the cost of a flip: the cost of changing a CP-net is the total number of flips performed in the cp-statements, each weighted according to the level of the relevant issue. More precisely: $\sum_x flip(x) \times (k+1-level(x))$, where x ranges over the issues, k is the number of levels in the CP-net, $flip(x)$ is the number of flips performed in cp-statements associated to x, and $level(x)$ is defined recursively as: $level(x) = 1$ if x is an independent issue; otherwise, $level(x) = i+1$ if all parents of x are in levels $\{1,\ldots,i\}$ and there is a parent in level i.

We can now state the **Combinatorial bribery problem**: We are given a profile (P,O) where P is a collection of n compact CP-nets with m binary issues and O is a total ordering of the m issues, a budget B, an outcome p, and bribing cost vector $\vec{Q}$ (each voter has its own cost, to be multiplied by the cost of the bribery actions according to the cost scheme). With this input, we want to know if there is a way for an outside actor to make p win in profile (P,O) with winner determination rule $D \in \{SM, OP, OV, OK\}$, by using bribery actions according to $A \in \{IV, DV, IV+DV\}$, and by paying according to scheme $C \in \{C_{\text{EQUAL}}, C_{\text{FLIP}}, C_{\text{LEVEL}}\}$ and bribing cost vector $\vec{Q}$, without exceeding B.

Table 1 shows the computational complexity of this problem, considering all possible combinations of winner determination rules, bribery actions, and cost schemes.

Starting from the first column, the NP-completeness result regarding SM with C_{EQUAL} is obtained via a reduction from the OPTIMAL LOBBYING (OL) problem [4]. Bribery with SM is instead easy with cost schemes C_{FLIP} and C_{LEVEL}: since we are working level by level, at each level we can select the agents to bribe by starting from the cheapest ones (according to $\vec{Q}$). The resulting preferred value for this issue can then be propagated in all CP-nets.

The second column shows SM with weighted voters. The results in this case are obtained via polynomial reductions from plurality-weighted-$bribery which was shown to be NP-complete in [6].

All the other entries of Table 1 relate to the extension to a combinatorial setting of the result by Faliszewski [5], which shows that plurality bribery in single issue elections with nonuniform cost functions is in P through the use of flow networks. The algorithm requires the enumeration of all possible elements of the candidate set as part of the construction of the flow network. In our model, the number of candidates can be exponential in the size of the input, so we cannot use that construction directly. We show that a similar technique works for OP and for all costs (except C_{LEVEL} when the briber can act on both dependent and independent issues) by considering only a polynomial number of candidates. For C_{LEVEL}, enumerating a polynomial number of cheapest (non-voted) alternatives for a voter becomes difficult making the flow-based approach non-applicable and, therefore, we conjecture this case is difficult.

The corresponding results for OV are obtained by the same line of reasoning and by noting that the worst outcome of a CP-net is the optimal outcome of the "reversed" CP-net, that is, the CP-net obtained by reversing the total orderings in all the cp-statements.

In OK, each agent gives its top k outcomes according to some linearization. When k is a power of two, it is possible to prove that if the top outcome is fixed, the next k outcomes must follow in some unique order. Therefore, it is possible to treat the top k outcomes as one bundle and to apply the flow-based approach in order to decide the cheapest bribery scheme to elevate the bundle that includes p into the winning set. We note that this is in contrast with the traditional bribery domain where the bribery problem for k-approval, when $k \geq 3$, is NP-complete even when all the bribery costs are equal [6].

4. FUTURE WORK

We are studying the open question left in our result table. We also plan to study non-binary domains, other scoring and voting rules, additional bribery actions that can also add dependencies, and the combination of weights with other voting rules.

5. ACKNOWLEDGMENTS

Nicholas Mattei was supported by NSF CCF-1049360 and NSF IIS-1107011, which, as part of the IJCAI-11 Doc. Consortium, funded his visit to Padova. This work has been partially supported by the MIUR PRIN 20089M932N project, "Innovative and multi-disciplinary approaches for constraint and preference reasoning."

6. REFERENCES

[1] K. J. Arrow, A. K. Sen, and K. Suzumura. *Handbook of Social Choice and Welfare.* North-Holland, Elsevier, 2002.

[2] C. Boutilier, R. I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *JAIR*, 21(1):135–191, 2004.

[3] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. *SOFSEM 2007*, pages 51–69, 2007.

[4] R. Christian, M. Fellows, F. Rosamond, and A. Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, 2007.

[5] P. Faliszewski. Nonuniform bribery. In *AAMAS08*, pages 1569–1572, 2008.

[6] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. How hard is bribery in elections? *JAIR*, 35:485–532, 2009.

[7] J. Lang and L. Xia. Sequential composition of voting rules in multi-issue domains. *Mathematical Social Sciences*, 57(3):304–324, 2009.