Preserving Consistency for Liquid Knapsack Voting

Extended Abstract

Pallavi Jain
IIT Jodhpur
pallavi@iitj.ac.in

Krzysztof Sornat
MIT CSAIL
sornat@mit.edu

Nimrod Talmon
Ben-Gurion University
talmonn@bgu.ac.il

ABSTRACT

Liquid Democracy (LD) uses transitive delegations in voting. In its simplest form, it is used for binary decisions, however its promise holds also for more advanced voting settings. Here we consider LD in the context of Participatory Budgeting (PB), which is a direct democracy approach to budgeting, most usually done in municipal budgeting processes. In particular, we study Knapsack Voting, in which PB voters approve projects, such that the sum of costs of projects each voter approves must respect the budget limit. We observe possible inconsistencies, as the cost of voter-approved projects may go over the budget limit after resolving delegations. We offer ways to overcome them by studying the computational complexity of updating as few delegations as possible to arrive—after following all project delegations—to a consistent profile.

KEYWORDS

Participatory Budgeting; Liquid Democracy; Knapsack Voting

ACM Reference Format:
Pallavi Jain, Krzysztof Sornat, and Nimrod Talmon. 2021. Preserving Consistency for Liquid Knapsack Voting: Extended Abstract. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

1 INTRODUCTION

In combinatorial Participatory Budgeting (PB) [1, 2, 4, 7, 12, 16]—used usually for deciding on the use of public funds in a municipality by direct democracy—voters provide their preferences over a set of projects, where each project has its own cost; the aggregation task is then to choose a project subset whose total cost does not exceed some given budget limit. In *Knapsack Voting* [13], voters provide their preferences as follows: each voter selects a subset of projects which total cost does not exceed the given budget limit. (Effectively, this means that each voter shall solve an instance of the Knapsack problem herself; Goel et al. [13] discuss advantages of Knapsack Voting for PB.)

In Liquid Democracy (LD) [3, 6, 14], voters can transitively delegate their decisions to other voters. The main merit of LD is the flexibility it gives voters, which also improves the scalability of decision making.

In this paper we investigate the use of LD in the context of Knapsack Voting for Participatory Budgeting (PB) [9]. While a simplest form of LD for Knapsack Voting would be to allow voters to delegate their whole ballot, here we are interested in even greater flexibility, and propose "project-wise delegations" (generally speaking, we follow the Pairwise Liquid Democracy approach [8] and the approach of Bloembergen et al. [5]): each voter v, for each project p, can choose between the following two options: (1) vote directly, deciding whether to approve p or disapprove p (i.e., whether to select p or not); or (2) delegate the decision on whether to approve p or disapprove p to another voter of her choice.

While such project-wise delegations indeed provide even greater voter flexibility, there is a risk of inconsistent ballots; consider the following example with 3 voters u, v, w, and 4 projects a, b, c, d, where voter u is positive towards approving project a but does not like project b. Then, u approves project a and disapprove project b. It is possible, that voter u is not sure about the project c due to the lack of information, knowledge, or some other reason. But, she trusts voter v for the project c. So, she delegates her vote for c to v; and will take the same decision as of v. It is possible that u does not find v competitive enough to take decision for the project d and trusts w more than v. So, she delegates her vote for d to w. Now, say that each project costs 1 and the budget limit is 2. If v decides to approve project c and also w decides to approve project d then effectively voter u would approve projects a, c, and d, thus in total she approves projects of cost 3, which is strictly above the budget limit. Thus, due to following the direct decisions and the delegations as expressed by voter u, we have that the vote of u violates the constraints of Knapsack Voting.

Our main aim in this paper is to explore possibilities of mitigating such possible inconsistencies. To this end, our approach is to look for the most delicate changes we can apply to the given votes to arrive to an instance in which such inconsistencies are avoided. We formulate our approach as the combinatorial problem of updating as few delegations as possible so that, after following all delegations, all votes satisfy the Knapsack constraint of Knapsack Voting. Figure 1 shows an example of solving an instance of Consistent Knapsack Voting (CKV) defined as follows.

> Consistent Knapsack Voting (CKV)
> **Input:** A set of projects P, cost function $c\colon P \to \mathbb{R}_{\geq 0}$, a budget B, an integer k, a set of voters V, a set of votes $\{L_v\colon v \in V\}$, where $L_v \in \{0, 1, V \setminus \{v\}\}^{|P|}$ and $L_v(p) \in V \setminus \{v\}$ means that a voter v delegates decision on p to $L_v(p)$.
> **Question:** Can we update at most k delegations to 0 or 1 so that the resultant instance, after following all transitive delegations, would result in all voters respecting the Knapsack constraint?

2 RESULTS

We analyze the parameterized complexity [10, 11] of CKV, for these parameters: the number $|V|$ of voters, the number $|P|$ of projects, the budget limit B, and the number of updates k (we refer to k also as the *solution size*), the number $D = \sum_{v \in V} |\{p \in P\colon L_v(p) \notin \{0, 1\}\}|$

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

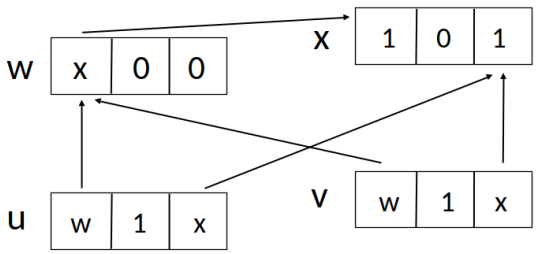

Figure 1: An instance of CKV**. There is a set of projects** $P = \{p_1, p_2, p_3\}$**, each of unit cost, and the budget limit is** $B = 2$**. There are voters** $V = \{u, v, w, x\}$**, whose votes are represented as vectors where the left entry corresponds to** p_1**, the middle entry to** p_2**, and the right entry to** p_3**; e.g., voter** u **delegates the decision on** p_1 **to** w**, approves** p_2**, and delegates the decision on** p_3 **to** x**. Note that if we follow the delegations then both** u **and** v **would violate the Knapsack constraints. If we update the delegation of voter** w **wrt.** p_1 **(i.e., change it so that** w **would disapprove** p_1**), however, then we will have a consistent instance, thus with** $k = 1$**, this is a yes-instance.**

of delegations, the maximum number of delegations a single voter can use and the maximum length of *a delegation chain*[1].

First, we offer some preprocessing methods:

(1) **Removing cycles:** indeed, there could be cycles (e.g., voter u delegates the decision on p to voter v, which, in turn, delegates the decision on p "back" to voter u (of course, there might be also cycles containing more than two voters); in the context of CKV, however, we can change all corresponding votes in a cycle to be disapprovals, as we are interested in minimizing the number of changes of delegations to have Knapsack-consistent ballots, so having disapprovals is always better for us than having approvals. So we can assume wlog. that there are no delegation cycles in our CKV instances.

(2) **Removing disapproval chains** A *disapproval chain* is a delegation chain ending in a disapproval; e.g., voter u delegates the decision on p to v, v delegates the decision on p to w, and w disapproves p; following the delegations, we will have disapprovals for all voters in a disapproval chain, for the corresponding project at hand. So, as having disapprovals is always better for us in the context of CKV, we can simply change all delegations in a disapproval chain to be disapprovals. So we can assume wlog. that there are no disapproval chains in our CKV instances.

Our main results are aggregated in the following theorem.

Theorem 1. *The following hold:*

- CKV *is* NP*-hard even if: each voter delegates to only one other voter,* $|P| = 4$*,* $B = 1$*, the maximum number of delegations in a vote is* 3*, the maximum number of approvals in a vote is* 1*, the costs of all projects are equal, the maximum length of a delegation chain is* 2*, the maximum in-degree in the delegation graph is* 3*.*
- CKV *is* W[2]*-hard wrt.* k *even if each voter can delegate to only one other voter.*
- CKV *can be solved in polynomial-time when every voter delegates at most one project.*
- CKV *can be solved in time*[2] $O^*\left(\binom{C}{k}\right) \leq O^*(2^C)$*, where* C *is the number of delegation components.*
- CKV *can be solved in time* $O^*(2^{|V|\cdot|P|})$*.*
- CKV *can be solved in* $O^*(2^{O(2^{|V|+\log|V|})})$ *time, hence the problem is* FPT *wrt.* $|V|$*.*
- *Unless Exponential Time Hypothesis [15] fails, there is no algorithm for* CKV *that runs in time* $2^{o(|P|+|V|+k+B)}$*.*
- CKV *can be solved in polynomial-time if all delegation chains are of length at most one.*
- *We can formulate* CKV *as an integer program with* C *binary variables, where* C *is the number of delegation components. (In the full version of the paper we report on computer simulation made using such a formulation.)*
- *Let* $\mathcal{I}$ *be an instance of* CKV *and let* $p \in [0, 1)$ *be a fixed constant. There exists a polynomial-time algorithm that returns a solution to* $\mathcal{I}$ *in which all voters exceed (additively) the Knapsack constraint by at most* $\max\{3\ln(2|V|), \sqrt{3\ln(2|V|)B}\}$ *with probability at least* p*. Note that in the case* $3\ln(2|V|) \leq B$ *it is a* 2*-approximation algorithm wrt. the Knapsack constraint.*

3 DISCUSSION AND OUTLOOK

Motivating by the desire to get the benefits of LD for PB, we considered the option of allowing per-project delegations for Knapsack Voting. Observing that with the additional flexibility given to voters, there could be inconsistencies by naively following delegation chains, we considered altering as few delegations as possible to solve such inconsistencies. We have shown that this is computationally very hard; but it can be done efficiently (1) if $|V|$ is small and for other restricted cases; (2) if we settle for approximation algorithms; and (3) if we are using ILP solvers. This opens the way to solve many instances efficiently. We believe that it is useful to allow fine-grained transitive delegations to PB as we view our results as showing that it is possible to enable it, at least in certain cases. The most pressing future research direction would be to develop and analyze other ways of solving such inconsistencies, such as: (1) having voters rank the projects she delegates so that we could go greedily over such rankings; (2) updating a set of delegations whose total cost is the minimum (and not its size, as we study here); (3) updating a set of delegations while not tackling only the roots of delegation chains, e.g., by minimizing the number of voters affected.

ACKNOWLEDGMENTS

Krzysztof Sornat was partially supported by the Foundation for Polish Science (FNP) within the START programme, the National Science Centre, Poland (NCN; Grant No. 2018/28/T/ST6/00366) and the Israel Science Foundation (ISF; Grant No. 630/19). Nimrod Talmon was supported by the Israel Science Foundation (ISF; Grant No. 630/19).

[1] An *approval delegation chain* for some project p is a sequence of voters, each delegates the decision on p to the next voter in the chain, where the last voter in the chain delegates the decision on p to some voter that approves p. Disapproval delegation chains can be removed in preprocessing.

[2] O^* hides factors that are polynomial in the input size.

REFERENCES

[1] Haris Aziz, Barton E. Lee, and Nimrod Talmon. 2018. Proportionally Representative Participatory Budgeting: Axioms and Algorithms. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2018)*. 23–31.

[2] Haris Aziz and Nisarg Shah. 2020. Participatory Budgeting: Models and Approaches. *CoRR* abs/2003.00606 (2020).

[3] Jan Behrens. 2017. The Origins of Liquid Democracy. *The Liquid Democracy Journal* 5, 2 (2017), 7–17.

[4] Gerdus Benade, Swaprava Nath, Ariel D. Procaccia, and Nisarg Shah. 2017. Preference Elicitation for Participatory Budgeting. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI 2017)*. 376–382.

[5] Daan Bloembergen, Davide Grossi, and Martin Lackner. 2019. On Rational Delegations in Liquid Democracy. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*. 1796–1803.

[6] Christian Blum and Christina Isabel Zuber. 2016. Liquid Democracy: Potentials, Problems, and Perspectives. *Journal of Political Philosophy* 24, 2 (2016), 162–182.

[7] Florian Brandl, Felix Brandt, Dominik Peters, Christian Stricker, and Warut Suksompong. 2019. Donor Coordination: Collective Distribution of Individual Contributions. http://brandlf.com/publications/donate. Online; accessed 15 February 2021.

[8] Markus Brill and Nimrod Talmon. 2018. Pairwise Liquid Democracy. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018)*. 137–143.

[9] Yves Cabannes. 2004. Participatory Budgeting: A Significant Contribution to Participatory Democracy. *Environment and Urbanization* 16, 1 (2004), 27–46.

[10] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2015. *Parameterized Algorithms*. Springer.

[11] Rodney G. Downey and Michael R. Fellows. 1995. Fixed-Parameter Tractability and Completeness I: Basic Results. *SIAM J. Comput.* 24, 4 (1995), 873–921.

[12] Rupert Freeman, David M. Pennock, Dominik Peters, and Jennifer Wortman Vaughan. 2019. Truthful Aggregation of Budget Proposals. In *Proceedings of the 2019 ACM Conference on Economics and Computation (EC 2019)*. 751–752.

[13] Ashish Goel, Anilesh K. Krishnaswamy, Sukolsak Sakshuwong, and Tanja Aitamurto. 2019. Knapsack Voting for Participatory Budgeting. *ACM Trans. Economics and Comput.* 7, 2 (2019), 8:1–8:27.

[14] James Green-Armytage. 2015. Direct Voting and Proxy Voting. *Constitutional Political Economy* 26, 2 (2015), 190–220.

[15] Russell Impagliazzo and Ramamohan Paturi. 2001. On the Complexity of k-SAT. *J. Comput. Syst. Sci.* 62, 2 (2001), 367–375.

[16] Nimrod Talmon and Piotr Faliszewski. 2019. A Framework for Approval-Based Budgeting Methods. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*. 2181–2188.