Enumeration and Exact Design of Weighted Voting Games[1]

Bart de Keijzer [a] Tomas Klos [b] Yingqian Zhang [b]

[a] *Centrum Wiskunde & Informatica, Amsterdam, The Netherlands*
[b] *Delft University of Technology, Delft, The Netherlands*

Introduction

In many multiagent settings, situations arise in which agents must collectively make decisions while not every agent is supposed to have an equal amount of influence in the outcome of such a decision. Weighted voting games (WVGs) are often used to model these kinds of situations: each agent is assigned a weight, and a decision is taken if the total weight of the coalition of agents in favor of the decision exceeds a given quota. Such rules are used for decision making in the US Electoral College, the IMF, the EU's council of ministers, and stockholder companies, among others.

The amount of influence an agent has in a weighted voting game is not the same as the (relative) number of votes she can cast. For example, if the quota is equal to the sum of all agents' weights, then, *irrespective of their weights*, all agents are equally powerful, at least in the sense that every agent's vote is required for the decision to go through. A variety of so-called *power indices* have been proposed to measure agents' power in WVGs, most notably the Banzhaf or the Shapley-Shubik index. In our paper, we propose the first exact algorithm for the 'inverse' problem of designing a weighted voting game in which the distribution of power among the agents (according to a given power index) is as close as possible to a given target vector of power indices. This is a very relevant problem when designing voting rules for the EU's council of ministers, for example [1], where the target vector gives each country an amount of influence proportional to its population size. Unfortunately, it is already intractable to calculate most power indices, let alone solve this, what we call Power index Voting Game Design (PVGD) problem. Nonetheless, we propose a method to solve this problem exactly, which relies on a new efficient procedure for enumerating weighted voting games of a given number of agents. One might imagine that an exponential algorithm is not so bad if it needs to be run just once (for every enlargement of the EU, or significant change of population distribution, for example).

Approach

Our approach to solving the inverse problem is to enumerate all possible WVGs of n agents, and to compute for each of these weighted voting games its power index. We can then output the game of which the power index is closest to the target vector of agent power indices. Unfortunately, it turns out that finding a good way of enumerating all weighted voting games is not straightforward.

A naive algorithm For one thing, we can not enumerate the games in the most intuitive representation, the *weighted form*. In this representation, for example, a 4-player majority voting game in which one player has double the weight of the other players, might be represented as $(3; 2, 1, 1, 1)$, where the quota is 3 since that is more than half the sum of the 4 players' weights (the last 4 numbers). Even though there are only finitely many WVGs for a given number of players, there are, as we show in our paper, infinitely many weighted

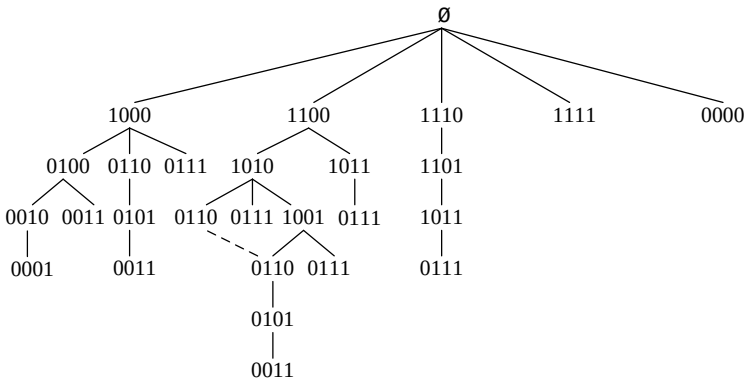
[1] This is an extended abstract of a paper originally published in the AAMAS 2010 Proceedings:
http://ifaamas.org/Proceedings/aamas2010/pdf/01%20Full%20Papers/07_05_FP_0238.pdf.

form representations for even a single WVG, making enumeration impossible in this representation. Fortunately, there are other, more appropriate representations, such as listing all the (finitely many) winning or losing coalitions of a game.

A first, naive, algorithm, then, lists all of the approximately 2^{2^n} monotone games as sets of minimal winning coalitions (MWCs)—winning coalitions from which no agent can be removed without making the coalition lose—and checks for each of those games whether it is a WVG, which it is if a weighted form for it can be found (in polynomial time [2]). If the game is a WVG, the algorithm calculates its power index and compares it to the target, eventually outputting the game whose power index is closest to the target.

Exponential improvement For an exponential improvement, we focus on the class of *canonical* WVGs, which are WVGs whose weight vectors are non-increasing. (For any WVG that is not a canonical WVG, there is a canonical one that can be obtained by just permuting the players.) Theorem 4 of our paper shows that the $\supseteq$-relation among the sets of MWCs describing games partially orders the set of canonical WVGs, while providing a least element. The theorem basically says: "Consider an arbitrary weighted voting game of n agents, and look at its list of MWCs. There is a MWC in this list, such that if we remove that MWC, we obtain a list of MWCs that represents yet another weighted voting game of n agents."

The figure below depicts this partial order on the set of 4-player WVGs, which has 27 elements (WVGs): each node in the graph is a WVG which has as its set of MWCs all coalitions labeling the nodes on the path from the game's node to the root. The poset is not a tree, which complicates our algorithm. The set of MWCs defining the 'lowest' of the three games at nodes labeled 0110, is a superset of *two* sets of MWCs that represent WVGs.



This partial order on the set of canonical WVGs (CWVGs) allows us to speed up the enumeration. We start by outputting the WVG with zero MWCs (the game at the root of the poset). Thereafter, we generate the MWC representation of all CWVGs with i MWCs using the set of CWVGs with $i-1$ MWCs, for increasing values of i, until we can find no further CWVGs. Also, we check that we do not output duplicate games (cf. the dashed line in the figure). This is the process by which the diagram in the figure was generated. For each game we output, we calculate the power index, and in the end we return the game with the power index closest to the target. The algorithm runs in $O^*(2^{n^2+2n})$ time, but is polynomial in the size of the output.

Conclusion

We now have an algorithm for solving the power index voting game design problem exactly. Usually, the number of players in such games is small, say 10 to 50. Currently, however, even 10 players is beyond the scope of our implementation. Our present work is focused on improving this implementation, as well as on improving the design of our algorithm, for example by investigating whether we can safely prune 'branches' of the partial order in the enumeration.

References

[1] Dennis Leech. Designing the voting system for the Council of the EU. *Public Choice*, 113, 2002.

[2] Uri Peled and Bruno Simeone. Polynomial-time algorithms for regular set-covering and threshold synthesis. *Discr. Appl. Math.*, 12, 1985.