The Cost Perspective of Liquid Democracy: Feasibility and Control

Shiri Alouf-Heffetz[1], Łukasz Janeczko[2], Grzegorz Lisowski[2], Georgios Papasotiropoulos[3]

[1]Ben Gurion University, Beersheba, Israel
[2]AGH University, Krakow, Poland
[3]University of Warsaw, Warsaw, Poland
shirihe@post.bgu.ac.il, ljaneczk@agh.edu.pl, glisowski@agh.edu.pl, gpapasotiropoulos@gmail.com

Abstract

We examine an approval-based model of Liquid Democracy with a budget constraint on voting and delegating costs, aiming to centrally select casting voters ensuring complete representation of the electorate. From a computational complexity perspective, we focus on minimizing overall costs, maintaining short delegation paths, and preventing excessive concentration of voting power. Furthermore, we explore computational aspects of strategic control, specifically, whether external agents can change election components to influence the voting power of certain voters.

1 Introduction

In a perfect democracy, every citizen would ideally have the time, knowledge, and means to actively participate in every political decision impacting the society. This ideal is hindered by various costs, such as the effort required to understand complex issues or the financial burden of frequent decision-making processes. Liquid Democracy addresses these challenges by allowing well-informed individuals to vote directly, while others (transitively) delegate their voting rights to trusted representatives. By requiring only a subset of the electorate to form opinions and vote, it reduces overall cognitive and operational costs. Originating in the 20$^{\text{th}}$ century (Tullock 1967; Miller III 1969; Green-Armytage 2015; Blum and Zuber 2016), Liquid Democracy has seen widespread implementation in recent years (Paulin 2020) and has since become a prominent research area within Computational Social Choice (Grossi et al. 2024). Despite its focus on reducing election-related costs, their explicit consideration remains unexplored. We address this by analyzing a natural Liquid Democracy framework operating within a budget constraint.

In a Liquid Democracy system costs can arise from various factors, such as the time or effort needed to understand the election topic or to identify participants aligned with their views for vote delegation. These costs broadly reflect preferences between delegation and direct voting. Alternatively, they could correspond to monetary incentives to encourage participation in decisions on complex issues, e.g., when voters prefer not to engage but recognize that some must for the collective good. Costs may also stem from the voters' expertise based on voting history or their profile and interests. Depending on the context, they can be specified by voters or inferred by the system.

Selecting trusted representatives under budget constraints extends beyond Liquid Democracy. The following examples demonstrate how our model and questions apply to other domains where being or trusting a representative incurs costs, aiming to minimize these expenses while meeting explicit trust requirements. In a sensor network, a subset of sensors may be designated as cluster heads to aggregate and transmit data to a central hub (Mishra, Gupta, and Gui 2019). These could be chosen to optimize communication and data-driven decision-making, while minimizing costs related to head designation, such as energy consumption and security concerns. In autonomous vehicle fleets, leadership roles in decision-making, like route planning and traffic management, incur costs due to the advanced processing and communication required in control vehicles. Trust from other vehicles is essential for seamless, reliable operation and safety (Awan et al. 2020). In decentralized blockchain networks, validators or miners validate transactions, ensuring network integrity and reducing computational costs for non-validator nodes (Gersbach, Mamageishvili, and Schneider 2022). Various implementations in cryptocurrency systems (see, e.g., Project Catalyst and Optimism) allow stakeholders to vote on protocol changes through representatives, rewarding the latter for their expertise and time spent understanding protocol details.

In Liquid Democracy literature, cost is often assumed to exist for differentiating voters' options, though not in a quantifiable manner. For instance, in the works of Escoffier, Gilbert, and Pass-Lanneau (2019, 2020) and Markakis and Papasotiropoulos (2021) voters prefer options that are less costly, but this preference is not numerically captured. In contrast, Armstrong and Larson (2021) and Bloembergen, Grossi, and Lackner (2019) explicitly consider costs albeit in a decentralized scenario. A centralized approach is explored by Birmpas et al. (2024), which involves rewarding voters to become representatives. While this work, like ours, focuses on selecting a budget-feasible solution, their objective is to maximize the probability of identifying a ground truth, whereas ours emphasizes voter representation.

The model we examine aligns also with research on approval voting for committee elections, especially where voters choose representatives among themselves. Refer to the works of Alon et al. (2011) and Kahng et al. (2018) for motivation and related literature. However, staying in line with the context of Liquid Democracy, we assume transitivity of trust, i.e., each voter trusts those whom their trustees trust. Our optimization criteria differ as well.

Our Contribution

In our work we explore decision-making scenarios modeled by a graph where vertices represent entities (or voters, in the terminology of Liquid Democracy) and edges denote trust relationships (potential delegations). Each voter is associated with a *voting cost* and a *delegating cost*. Importantly, our results also apply when delegating costs are zero – a scenario relevant to several of the examples presented earlier. We work towards selecting voters to cast ballots under a constraint on the costs. Crucially, voters may be asked to vote despite their reluctance not only if they have lower voting costs than others but also if they are widely trusted as representatives. In line with theoretical works and common practices in Liquid Democracy, we assume transitivity of trust: If a voter votes on behalf of another, the latter is considered as satisfied as long as there is a directed path of approved representatives connecting them. To incorporate this we require that all voters are represented, either directly or through paths of trust relationships. In the first part of our work, we design and analyze from the perspective of computational complexity mechanisms that:

> *Minimize the total voting and delegating cost, while ensuring that every voter is either selected to vote or has a chain of trust leading to someone who is selected.*

On top of this feasibility constraint, still under the objective of cost minimization, and towards suggesting intuitively better solutions, we additionally aim (1) *to ensure each delegating voter has a reasonably short delegation path to a representative*, mitigating diminishing trust with path length, and (2) *to prevent excessive concentration of voting power*, maintaining democratic legitimacy. Both desiderata have been identified as crucial and analyzed in prior works on Liquid Democracy, though in different contexts (Brill et al. 2022; Gölz et al. 2021; Kling et al. 2015).

In the second part of our work we define and algorithmically study a family of control problems related to the aforementioned constraint on voters' voting power. Control of elections is a prevalent area of Computational Social Choice (Faliszewski and Rothe 2016), recently garnered attention in Liquid Democracy settings (Bentert et al. 2022; Alouf-Heffetz et al. 2024). Unlike the literature focusing on controlling outcomes, since voters' preferences over them are not explicitly present in our framework, we consider scenarios where:

> *An external agent favoring a certain voter, aims to manipulate the instance to enhance their power.*

This marks a conceptual contribution, which could pave the way for further strategic studies in Liquid Democracy and beyond, shifting attention from just election outcomes.

Our results show that if each voter specifies at most one individual as an acceptable representative, optimally selecting voters to cast ballots is computationally feasible under all of the examined constraints. However, allowing voters to approve more representatives makes most of the related problems hard. A similar pattern is observed in the examined control problems. Despite the apparent similarity in some of the statements of our findings, the proofs themselves vary significantly. Furthermore, we designed the proofs attempting to be broadly applicable, allowing us to extend the results to related computational problems which are well-motivated within the examined setting.

Due to space constraints, the full proofs of some of our results are omitted.

2 Model and Definitions

The core component of the setting that we study is a directed graph $G(N, E)$, which we call a *delegation graph*, where the set N of vertices represent *voters* and the set E of edges show to whom a voter might delegate. Specifically, an edge from a voter $i \in N$ to a voter $j \in N$ signifies that i approves j as their (immediate) representative. We will refer to the tuple $(G(N, E), v, d)$ as an instance of a Liquid Democracy scenario under costs, or a *cLD election* in short. An important parameter of a cLD election is the *maximum out-degree* of vertices of the corresponding delegation graph, which we denote by Δ. We will call *predecessors* of a vertex $i \in N$ all the vertices that have a directed path to i. Moreover, each voter $i \in N$ is associated with two costs: $d(i)$ for delegation and $v(i)$ for voting. We note that both values may be zero and that there is no inherent ordering between them.

A *delegation function* is a mechanism that takes a cLD election $(G(N, E), v, d)$ as input, and returns a set $C \subseteq N$ representing voters who will cast a ballot, together with a set $D \subseteq E$ of exactly one outgoing edge for each vertex not in C denoting the selected delegations. We call the pair (C, D) a *solution* under a delegation function on $(G(N, E), v, d)$. Under a delegation function, we refer to the selected subset of voters C as the set of *casting voters* and as *delegating voters* to the voters from $N \setminus C$. The first computational problem that we consider follows.

DELEGATE REACHABILITY	
Input	A cLD election $(G(N, E), v, d)$ and a budget parameter $\beta \in \mathbb{N}^*$.
Question	Does there exist of a delegation function of solution (C, D) satisfying: (i) the *cost constraint*: $\sum_{i \in C} v(i) + \sum_{i \notin C} d(i) \leq \beta$, (ii) the *reachability constraint*: for every vertex $i \notin C$ there is a directed path to a vertex in C using only edges in D.

When β is not part of the input of the considered problem, a solution, and in turn a delegation function, is said to be *cost-minimizing* if it satisfies the cost constraint for the minimal value of β for which reachability is also met.

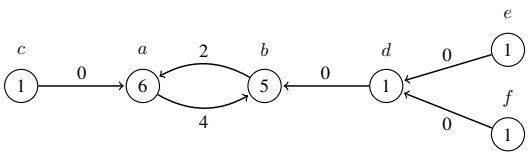

Figure 1: An example of a delegation graph on 6 voters, namely $\{a, b, c, d, e, f\}$. Numbers inside the vertices indicate voting costs, while those on an edge (u, v) show the delegating cost of the voter corresponding to u.

In principle, a delegation function can be viewed as a way to partition N into casting and delegating voters, though it does not yet ensure that delegation cycles are avoided. However, a solution satisfying the reachability constraint not only involves the specification of casting voters but also designates a specific casting voter, or a *representative*, for each delegating one, by following the paths to casting voters indicated by D. Under a delegation function of the solution (C, D), we denote by R_j the set of delegating voters who have a path to a casting voter j through edges in D, i.e., those who will ultimately be represented by j. Consequently, j will vote with a voting weight, or *voting power*, of $|R_j|+1$, representing the voters in R_j along with themselves.

Example 2.1. *Consider the delegation graph in Figure 1. There are 6 voters, a to f, for whom casting a ballot is more expensive than delegating. To ensure that a path exists from each voter to a casting one, any set including a or b could be selected to cast a ballot. However, if cost minimization is the (sole) objective, setting $C = \{a\}$ is necessary and it comes at a total cost of 8: 6 from a's voting cost, 2 from b's delegating cost, while the rest delegate for free. Under this solution, a holds a voting power of 6.*

In practice, and setting aside the costs for a moment, an electorate in a Liquid Democracy scenario under the discussed model can be partitioned into two groups: Voters who prefer casting a ballot and those who do not. Essentially, a delegation function satisfying the cost and reachability constraints is responsible for determining which voters who feel like they do not want to cast a ballot should be required to do so. This could be due to it being easier for them to become voters rather than others, or for the benefit of society, meaning they are widely approved as representatives by others (either directly or through a path of trust relationships). A natural question then arises regarding what happens to those voters who prefer to cast a ballot rather than delegate.[1] Importantly, the objective of cost minimization in our setting suggests the expected outcome: voters who are willing to vote will be asked to do so.

Remark 2.2. *A cost-minimizing solution includes in the selected set of casting voters every voter who prefers to cast a vote rather than delegate, i.e., satisfies $\{i \in N : v(i) < d(i)\} \subseteq C$. This also applies to voters who do not express their consent for being represented by others.*

[1]Voters who prefer to cast a ballot need not be asked to specify delegating costs at all, however, for uniformity, we describe the setting generally, as this observation does not affect our results.

3 Feasible Delegation Functions

We begin our analysis on the computational complexity of finding feasible delegation functions by showing that DELEGATE REACHABILITY is tractable.

Theorem 3.1. DELEGATE REACHABILITY *is solvable in polynomial time.*

Proof Sketch. We present a greedy method for computing a cost-minimizing solution. We start by constructing the decomposition $\mathcal{D}(G)$ of the delegation graph G into strongly connected components. Recall that a subgraph of a directed graph is called a strongly connected component (SCC) if, for every pair of vertices u and v within it, u is reachable from v and vice versa. The *decomposition* of a graph G into SCCs is a directed acyclic graph having one vertex for every maximal strongly connected component of G and a directed edge from the vertex for component S to the vertex for component S' if G contains an edge from any vertex in S to any vertex in S' (refer to the textbook by Dasgupta, Papadimitriou, and Vazirani (2006) for a more extensive discussion on SCC decompositions). We notice that in each feasible solution of DELEGATE REACHABILITY, it is necessary to select at least one voter as a casting voter from each set of voters that belong to the component corresponding to a sink of $\mathcal{D}(G)$; note that $\mathcal{D}(G)$ has a sink as it is acyclic. First, identify all voters whose voting cost is strictly less than their delegation cost. These voters are designated as casting voters. Then, for each sink of $\mathcal{D}(G)$ that has not yet been assigned at least one casting voter, it is sufficient to nominate the voter who minimizes the sum of their voting cost plus the delegation costs of the other voters in the sink. □

In many cases, it may be impossible to find a feasible delegation function that satisfies both the cost and the reachability constraints. Instead of deeming such scenarios indomitable immediately, we suggest first exploring a natural way or circumventing this issue, already addressed in other Liquid Democracy frameworks as well (Colley, Delemazure, and Gilbert 2023; Campbell et al. 2022; Markakis and Papasotiropoulos 2021). One way to expand the solution space is to allow for a given number, α, of abstainers, requiring feasibility only after excluding them. To expand further, as discussed in Section 1, ideally, in the examined model voters uncomfortable with casting a ballot should be assigned to a casting voter through a path of trust relationships. However, if this proves impossible, and with a slight abuse of notation, a solution would be to permit a small number of voters to remain unrepresented. Unfortunately, with this additional degree of flexibility, the corresponding variant of DELEGATE REACHABILITY becomes computationally hard.

Theorem 3.2. *Allowing for α abstainers,* DELEGATE REACHABILITY *is NP-complete.*

Incorporating the possibility of abstainers into the model also aligns well with the line of work on control problems discussed in Section 1 and which we analyze in Section 4. Imagine a controller (not necessarily a malicious one this time) aiming to alter the election components to turn a negative instance of DELEGATE REACHABILITY into a positive one, thereby ensuring that the election is not deemed

useless. By selecting a set of abstainers to make certain that DELEGATE REACHABILITY admits a valid solution, the controller (or election organizer) can essentially determine which voters to exclude to make the instance feasible. For completeness, we note that with a slight modification to the proof of Theorem 4.3 we can also prove hardness for the problem of adding voters to ensure feasibility.

We conclude the section by highlighting that Theorem 3.2 holds for a value of α that depends on the parameters of the input instance. However, if α is fixed, then a generalization of the result of Theorem 3.1 is possible. Specifically, one can easily apply, in polynomial time, the algorithm presented after the deletion of any possible set of at most α voters.

Proposition 3.3. *With a constant number of abstainers,* DELEGATE REACHABILITY *is solvable in polynomial time.*

In essence, Theorem 3.1 asserts that the existence of a *sufficiently good* delegation function which meets basic cost and representation requirements can be efficiently decided. In what follows, we will focus on identifying necessary and sufficient conditions for achieving intuitively *significantly better* solutions, referring mainly to sets of casting voters that satisfy additional requirements.

Refined Feasible Delegation Functions

We will focus on instances that admit a feasible solution under DELEGATE REACHABILITY. Specifically, when sets of casting voters that meet both cost and reachability constraints exist, one delegation function may be seen as superior to another based on various metrics, with two natural requirements being (1) aiming for solutions that create short paths from delegating voters to casting voters, and (2) aiming for solutions that propose casting voters which will not acquire excessively large voting power.

Objective (1) follows the principle that the longer the proposed delegation path from a delegating voter v to a casting voter, the less v trusts their final representative. It is natural to expect (and always assumed in Liquid Democracy) that a voter trusts their immediate successor, and also the successor's successor. However, this plausibly comes with a reduction in trust (Boldi et al. 2009; Armstrong, Alouf-Heffetz, and Talmon 2024; Brill et al. 2022). Hence, the shorter the delegation paths, the better.

Objective (2) has been extensively discussed and analyzed in Liquid Democracy frameworks. We refer to Gölz et al. (2021) and Kahng, Mackenzie, and Procaccia (2021) for comprehensive analyses on the inherent danger that voters with high power can bring upon election instances, and the threat this poses to the democratic nature of the system.

Example 3.4. *Take the instance from Figure 1. Notice that if the budget allows, selecting $\{a, d\}$ as casting voters could be better than choosing $\{a\}$, as it splits the voting power between two casting voters. However, if the goal is to ensure that delegating voters have short distances to their representatives, setting the set of casting voters to $\{b\}$ is better than $\{a\}$, as then all voters would have a path of length at most 2 to their representative, budget permitting.*

We now define the computational problems aimed at refining the solutions of DELEGATE REACHABILITY.

BOUNDED MAX LENGTH / BOUNDED POWER	
Input	A cLD election $(G(N, E), v, d)$, a budget $\beta \in \mathbb{N}^*$ and a parameter $\ell \in \mathbb{N}^*$.
Question	Does there exist a delegation function of solution (C, D) which satisfies the cost and reachability constraints while ensuring that: ▶ Each path from a delegating voter to its nearest voter in C via edges from D is of length at most ℓ, for BOUNDED MAX LENGTH problem, ▶ The maximal voting power does not exceed ℓ, for BOUNDED POWER problem.

Henceforth, we will pay particular attention to delegation graphs with $\Delta = 1$. Conceptually, this scenario represents the first step away from direct democracy, with $\Delta = 0$ where all voters cast their own ballots without representation. In contrast, $\Delta = 1$ resembles forms of classic representative democracy, which is of course lacking the transitivity in ballots, while a priori partitioning voters into those who will represent and be represented. Technically, as we will demonstrate, the examined problems are computationally hard for instances with $\Delta = 2$. Therefore, exploring the restricted yet well-motivated case of $\Delta = 1$ is a natural first step toward characterizing polynomially solvable instances.

A directed tree, i.e., a directed graph the underlying undirected graph of which has a tree structure, is *upwards-directed* if there is a vertex that can be specified as its *root* or *sink*, and all other vertices have a directed path leading to it. It is easy to see that a delegation graph with $\Delta = 1$ consists of a disjoint union of (weakly) connected components, each containing at most one directed cycle. If a cycle exists, its vertices have no outgoing edges towards vertices outside the cycle. Thus, each component can be viewed as a graph with at most one cycle, whose removal would decompose the graph into a forest of upwards-directed trees. With this observation, we obtain the following tractability result through a dynamic programming procedure.

Theorem 3.5. *For $\Delta \leq 1$,* BOUNDED MAX LENGTH *is solvable in polynomial time.*

In contrast to DELEGATE REACHABILITY, moving to instances where Δ is larger than 1, even for cases where the maximal out-degree of voters is 2, BOUNDED MAX LENGTH becomes computationally hard.

Theorem 3.6. *For $\Delta > 1$,* BOUNDED MAX LENGTH *is NP-complete for every fixed $\ell \geq 2$.*

Proof Sketch. We show that the problem we consider is NP-hard by reducing from 3-SAT. We first show the claim for $\ell = 2$. Take a 3-CNF formula φ with the set of variables $X = \{x_0, \ldots, x_m\}$ and the set of clauses $\mathcal{C} = \{C_0, \ldots, C_l\}$. We will also represent each clause C_j as $\{L_j^1, L_j^2, L_j^3\}$, where each L_j^i corresponds to a different literal in C_j, for $i \in \{1, 2, 3\}$. Let us construct what we call an *encoding* of φ. Now, for each clause $C_j \in \mathcal{C}$ we add three voters, i.e., a *clause voter* v_{C_j}, as well as two *clause dummy* voters d_{C_j}, d'_{C_j} corresponding to C_j. Additionally, for each

variable $x_i \in X$, we add two *literal voters* corresponding to x_i and $\neg x_i$, namely voters v_{x_i} and $v_{\neg x_i}$. For every clause C_j, we construct edges, i.e., possible delegations, from the clause voter v_{C_j} to the clause-dummy voters d_{C_j}, d'_{C_j}. Also there are edges, i.e., possible delegations, from d_{C_j} to the literal voter corresponding to L^1_j and from d'_{C_j} to the literal voters that correspond to L^2_j and to L^3_j. Finally, for every $x_i \in X$, we add the following pair of edges: from v_{x_i} to $v_{\neg x_i}$ and from $v_{\neg x_i}$ to v_{x_i}. To complete the construction, we set voting costs to 1 and delegating costs to 0 and let $\beta = |X|$ and $\ell = 2$. Note that it holds $\Delta = 2$ and that the encoding of φ consists of $2|X| + 3|\mathcal{C}|$ voters and the shortest path from each clause voter to a literal voter is of length exactly 2.

Assume φ is satisfiable. Then, for each pair of literal voters v_{x_i} and $v_{\neg x_i}$, we take v_{x_i} as a casting voter if x_i is true in the considering satisfying assignment of φ, and $v_{\neg x_i}$ otherwise. This selection is budget feasible and ensures that each clause voter is at a distance of exactly two from a casting voter. Dummy and literal voters are also at a distance of at most two each. Conversely, consider the case that φ is unsatisfiable. It holds that any feasible solution for the encoding of φ requires choosing exactly one from each pair of literal voters, due to the budget constraint, and also that only literal voters can be casting voters. This leads to some clause voters being three steps away from a casting voter, violating the path length constraint.

We note that the construction can be modified to work for any larger value of ℓ, by increasing the distance between clause and literal voters, from 2 to ℓ. □

We next examine the problem of selecting representatives with the constraint being an upper bound on the maximal voting power of casting voters. While the obtained results are analogous to those for BOUNDED MAX LENGTH (being polynomially solvable for $\Delta \leq 1$ and NP-complete otherwise), the specifics of the proofs significantly differ.

Theorem 3.7. *For $\Delta \leq 1$,* BOUNDED POWER *is solvable in polynomial time.*

Proof Sketch. Note that in the case of $\Delta = 0$ the problem is trivial. We show that the claim holds by providing a dynamic programming algorithm. For this we define $dp[v, i, k]$ as the minimum cost of a (partial) solution that gives a voting power upper bounded by k to v and upper bounded by ℓ to all other casting voters of the considered subgraph, among the following vertices in G: v, its first i incoming neighbors and all of their predecessors. This is justified by the fact that vertices that will be considered next can increase the voting power of v but not the voting power of voters that have been already clasified as casting. We will compute a value for $dp[v, i + 1, k]$ based on $dp[v, i, \cdot]$ and $dp[v_{i+1}, p(v_{i+1}), \cdot]$, where v_{i+1} is the $(i+1)^{\text{th}}$ in-neighbor of v and $p(v_{i+1})$ is the number of its in-neighbors. If G is an upwards-directed tree, then the voter corresponding to the root should be nominated as a casting voter, among others, and a similar argument can be stated for arbitrary graphs satisfying $\Delta = 1$.

The values of the table can be computed by a bottom-up approach. Observe that the size of the table is polynomial in the input and it can be shown that the computation of its values can be also done efficiently. □

As in the BOUNDED MAX LENGTH, transitioning from a single accepted delegate per voter to allowing the approval of more potential representatives significantly increases the computational complexity of BOUNDED POWER.

Theorem 3.8. *For $\Delta > 1$,* BOUNDED POWER *is NP-complete for every fixed $\ell \geq 4$.*

Proof Sketch. For clarity, we first outline the proof for the case of $\ell = 4$. We prove NP-hardness by reducing from VERTEX COVER on 3-regular graphs. Considering such an instance $\mathcal{I}$ on ν vertices, we create the following encoding of $\mathcal{I}$ into an instance of BOUNDED POWER problem: First, for every vertex of $\mathcal{I}$, we construct a *vertex voter* corresponding to v. Moreover, for every edge e of $\mathcal{I}$ we construct an *edge voter* corresponding to e. Furthermore, if an edge e of $\mathcal{I}$ is between vertices i, j, we construct edges, i.e., possible delegations, from the voter corresponding to e to the voters who correspond to i and to j. We finally add a set D of ν *dummy voters*, who will vote at a cost of 0. For each vertex voter we add an edge to exactly one dummy voter in a way that each voter from D has exactly one incoming edge from vertex voters. Finally, for each voter v in D we add two additional voters, who are willing to delegate to v. We set the rest voting costs to 1 and all delegating costs to 0 and also we set $\beta = k$, where k is the decision parameter in $\mathcal{I}$, and $\ell = 4$. Note that $\Delta = 2$.

If $\mathcal{I}$ is a positive instance, we construct a budget feasible delegation function having the vertex voters corresponding to vertices in the cover along with dummy voters as casting voters. Then, each edge voter is represented by a vertex voter selected to cast a ballot, while the non-selected vertex voters will be represented by the dummy voters. Voting power of casting voters is bounded by 4. Supposing that $\mathcal{I}$ is a negative instance, a selection of k vertex voters as casting is not sufficient to cover all edge voters. This leads to dummy voters having to represent some edge voters, which causes the voting power to exceed the bound of $\ell = 4$. Hence, no feasible delegation function exists. In turn, the encoding of $\mathcal{I}$ is a negative instance of BOUNDED POWER as well.

Increasing the number of voters approving each dummy voter from 2 to $\ell - 2$, and applying the same arguments, proves the theorem for any value of ℓ. □

A further natural restriction on delegation functions involves averaging the lengths of paths to casting voters. While BOUNDED MAX LENGTH can be seen as having an egalitarian constraint, restricting the *sum of path lengths* is a utilitarian one. This new restriction balances BOUNDED MAX LENGTH and BOUNDED POWER, allowing a casting voter to represent either multiple voters nearby or a few farther away. The proof of Theorem 3.8 immediately establishes the hardness of BOUNDED SUM LENGTH, which asks whether a delegation function of solution (C, D) can satisfy cost and reachability constraints while keeping the sum of path lengths via D to each casting voter within a given bound ℓ. Moreover, Theorem 3.5 can be easily adapted to address this restriction.

Proposition 3.9. *For $\Delta > 1$,* BOUNDED SUM LENGTH *is NP-complete for every fixed $\ell \geq 4$, but for $\Delta \leq 1$ it is solvable in polynomial time.*

4 Strategic Control

We now begin our exploration of algorithmic questions concerning strategic control by an external (malicious) agent, referred to as the *controller*. In classic literature of election control (refer for instance to the work of Faliszewski and Rothe (2016)), the controller is typically considered able to manipulate components of the election such as the set of voters or candidates to achieve a desired outcome, under a prespecified voting rule. However, in our model presented in Section 2, which aligns with most studies in Liquid Democracy, we have abstracted away from the actual voting procedure and our focus has been on determining which voters will cast a ballot and consequently on the extent of their voting power; without considering the specifics of their votes. Therefore, unlike traditional election control studies and questions where the controller attempts to influence the final outcome, in our work the controller cannot take advantage of voters' preferences or final ballots, as these details are beyond our scope. Instead, the controller is allowed to influence only the delegation process and the specification of casting voters and their power.

We introduce a new class of control problems that integrates seamlessly with our model: The controller aims to manipulate the delegation process toward affecting the voting power of a designated voter. We capture this through the concept of a *super-voter*, defined as the casting voter with the highest voting power, after applying a specified delegation function.[2] We assume that the controller does not ex-ante know the precise delegation function that will be used, but only that it will be cost-minimizing while satisfying the reachability constraint, remaining consistent with our work's focus. Consequently, we say that the controller's goal is to ensure that their preferred voter becomes the (sole) super-voter, under every possible delegation function that meets these conditions.

CONTROL BY ADDING/DELETING VOTERS (CAV/CDV)	
Input	A cLD election $(G(N, E), v, d)$, a budget $\beta \in \mathbb{N}^*$, a parameter $k \in \mathbb{N}^*$ and ▶ a partition of N into registered (N_r) and unregistered (N_u) voters, and a designated voter $x \in N_r$, for CAV. ▶ a designated voter $x \in N$, for CDV.
Question	Does there exist a way to add up to k voters from N_u, in CAV, or to delete up to k voters, in CDV, so that x is the (sole) super-voter in the resulting election, under every cost-minimizing delegation function which satisfies the reachability constraint?

Example 4.1. *Take the instance from Figure 1. There, the cost-minimizing solution selects a as a casting voter, but a malicious external agent preferring b as the super-voter could achieve this by deleting a. Suppose that there is also a set X of unregistered voters (not depicted in Figure 1). Regardless of their costs or which voters they approve of, the controller cannot make b the unique super-voter under every feasible cost-minimizing solution by adding any subset of X.*

Theorem 4.2. CONTROL BY ADDING VOTERS *is NP-hard.*

We now move to the case where the controller is able to remove some voters from participating in the election. As in the case of adding voters, the relevant computational problem is NP-hard, although it requires different reduction.

Theorem 4.3. CONTROL BY DELETING VOTERS *is NP-hard.*

Proof Sketch. We show the hardness of CONTROL BY DELETION by reducing from CLIQUE. Given an instance of CLIQUE (G, k) we create an instance $\mathcal{I}'$ of the considered election control problem that contains the set of voters E containing one voter of delegating cost equal to 0 and voting cost equal to 1, for each edge in G. Also, we have a set of voters V containing one voter of delegating cost equal to 0 and voting cost equal to 1, for each vertex in G. Then, we take the set D of $(n - k) + (m - \binom{k}{2}) + 1$ voters, each of delegating cost equal to 0 and voting cost equal to 1 and special voters x, being the designated by the controller voter and $y_1, y_2, \ldots, y_k, y_{k+1}$, with voting cost equal to 0 and delegating cost equal to 1. Then, for each edge e that is incident to vertices u and v in G, we add two possible delegations: one from the voter of E that corresponds to e towards the voter of V that corresponds to v and one from the voter of E that corresponds to e towards the voter of V that corresponds to u. Then, there is an edge from each voter in V to every voter in $Y = \{y_1, y_2, \ldots, y_{k+1}\}$ and an edge from each voter in D to voter x. Finally say that the upper bound on the number of voters that can be deleted equals k.

Notice now that if there exists a k-clique $\mathcal{C}$ in G, then deleting voters corresponding to vertices in $\mathcal{C}$ ensures that the voters corresponding to the edges in $\mathcal{C}$ do not delegate their votes to any voter from Y in any cost-minimizing delegation function. Hence, x becomes a super-voter. Conversely, if a k-clique does not exist in G, then after deleting any k voters from the instance, sufficiently many voters corresponding to edges of G will delegate to a voter from Y under some delegation function. Hence, x cannot be guaranteed to be the only super-voter. □

Other strategic control problems can be defined in a similar vein. For instance, one might focus on adding or removing edges (i.e., possible delegations), instead of voters, to ensure that the preferred voter becomes the sole super-voter. The proofs from Theorems 4.2 and 4.3 extend to these problems with only minor adjustments. Furthermore, aligning with prior research on strategic control of classic elections (Hemaspaandra, Hemaspaandra, and Rothe 2007; Faliszewski, Hemaspaandra, and Hemaspaandra 2011), one might also explore the *destructive* counterparts of CAV and CDV. Specifically, this involves performing alterations to the voters' set towards *preventing* a specific voter from being the unique super-voter under every cost-minimizing delegation function that satisfies the reachability constraint. For these problems the proofs of Theorems 4.2 and 4.3 apply directly.

[2]The term has been used in earlier works (e.g., by Kling et al. (2015)) to refer generally to voters with a large share of incoming delegations, which is (slightly) different from how we use it here.

Proposition 4.4. *The variants of* CONTROL BY ADDING/DELETING VOTERS *where the controller can add or delete edges of the delegation graph are NP-hard. Moreover, the destructive variants of* CONTROL BY ADDING/DELETING VOTERS *are NP-hard as well.*

Following our approach in Section 3, we now turn to the examination of cLD elections where the maximal out-degree of vertices in the input delegation graph is upper bounded by 1. As one would expect, polynomial solvability now holds, but interestingly, the procedures we propose for solving the control problems differ completely from those used for proving Theorems 3.5 and 3.7.

Theorem 4.5. CONTROL BY ADDING VOTERS *is solvable in polynomial time if* $\Delta \leq 1$.

Proof Sketch. For $\Delta = 0$ it suffices to see that no voter may delegate, so x can be the unique super-voter only if $|N| = 1$. For $\Delta = 1$, we must first verify that x is a casting voter in every cost-minimizing delegation function; if not, additional voters cannot alter this, meaning the instance has no feasible solution. This verification can be done in polynomial time. We first convert x into the root of a tree by removing its outgoing edges, if any and only if the voting cost of x is higher than its delegating cost. Otherwise, there could be a delegation function where x delegates. We then aim to increase the voting power of x by adding unregistered voters who will delegate to x in any cost-minimizing delegation function. These voters are identified based on their costs as well as the existence of paths to x or to registered voters who will definitely delegate to x. We use a greedy algorithm to add these voters in layers: starting with those directly connected to x or to voters that x will represent, and iteratively including more until the budget k is reached. □

A greedy strategy can also identify which voters can be deleted to ensure a preferred voter becomes the only super-voter under any cost-minimizing delegation function.

Theorem 4.6. CONTROL BY DELETING VOTERS *is solvable in polynomial time if* $\Delta \leq 1$.

Continuing from Proposition 4.4 our discussion on strategic control problems involving the alteration of possible delegations, we note that the positive results for instances of $\Delta \leq 1$ are also applicable here. For the addition of edges, the proof is analogous to the proof of Theorem 4.5. For the deletion of edges, one must consider each potential casting voter y that has a voting power at least as high as that of the designated voter x, under at least one cost-minimizing delegation function and then delete iteratively the incoming edges to y, in descending order of the number of voters who would lose their paths to y as a result. This continues until y's voting power is reduced to below that of x.

Proposition 4.7. *The variants of* CONTROL BY ADDING/DELETING VOTERS *where the controller can add or delete edges of the delegation graph are solvable in polynomial time if* $\Delta \leq 1$.

In this section we focused exclusively on controlling the power of casting voters, however, the power of delegating voters should not be underestimated. As highlighted by Zhang and Grossi (2021) and Behrens et al. (2021), delegating voters can also control a substantial number of votes, impacting the election outcome. For example, a casting voter with direct delegations from $p - 1$ voters is intuitively more powerful than one whose voting power of p is achieved through an intermediary receiving exactly $p - 2$ delegations, as the latter's power depends entirely on the intermediary. Problems of ensuring a preferred voter gains multiple delegations, even without casting a ballot, naturally arise. Our results extend to such scenarios as well, with proofs being analogous.

5 Conclusion

Our work focused on computational problems related to determining feasible delegation functions and controlling the voting power of participants. Specifically, we first addressed questions around the existence of sufficiently good delegation functions (under cost and reachability constraints) considering well-established desiderata from the Liquid Democracy literature. We then concentrated on controlling the voting power of a preferred casting voter by adding or deleting voters to achieve the controller's goal. Our results provide a comprehensive understanding of the tractability landscape of these problems' families. For most of the considered problems, tractability is characterized (unless P=NP) by allowing each voter to approve at most one other voter as a potential representative.

Even beyond the Computational Social Choice context, the related graph-theoretic problems are of significant interest, making them a promising algorithmic direction with respect to, e.g., approximation or parameterized algorithms. Our results could serve as a starting point for analyzing the parameterized complexity of the examined problems. Indicatively, the hardness results from Section 3 show para-NP-hardness for ℓ and the positive results can be extended to parameterize by the number of voters with an out-degree greater than one and their maximum out-degree. Proposition 3.3 is also an XP[α] algorithm, and Theorem 4.3 is also a W[1]-hardness result for the number of voters that can be deleted. We also underline that our hardness results hold for restricted classes of delegation graphs, such as layered directed graphs with a few layers or even directed bipartite graphs. Moreover, the results on CAV and CDV hold even under additional restrictions on path lengths or voting power, along with the reachability constraint, in the formulation of these problems.

Our results can be complemented by simulating how factors like path length, voting power bounds, and budget constraints impact the frequency of feasible solutions or how one parameter influences others. Exploring how edge existence probabilities, cost distributions, or the number of voters impact feasibility in synthetic datasets could also be insightful. Similarly, considering costs, direct voting is democratically optimal but expensive, while (unconstrained) DELEGATE REACHABILITY offers much cheaper solutions. Solutions with constraints, such as bounded voting power or lengths of delegation paths, fall in the middle. Comparing these costs could reveal the savings of Liquid Democracy over direct voting, along with the additional costs introduced by constraints aimed at yielding intuitively better solutions.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 101002854). Georgios Papasotiropoulos is supported by the European Union (ERC, PRO-DEMOCRATIC, 101076570). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. The authors thank Piotr Faliszewski and Piotr Skowron for their helpful feedback and discussions.

References

Alon, N.; Fischer, F.; Procaccia, A.; and Tennenholtz, M. 2011. Sum of us: Strategyproof selection from the selectors. In *Proceedings of the Conference on Theoretical Aspects of Rationality and Knowledge*, 101–110.

Alouf-Heffetz, S.; Inamdar, T.; Jain, P.; Talmon, N.; and Hiren, Y. M. 2024. Controlling delegations in liquid democracy. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 2624–2632.

Armstrong, B.; Alouf-Heffetz, S.; and Talmon, N. 2024. Optimizing viscous democracy. *arXiv preprint arXiv:2405.06698*.

Armstrong, B.; and Larson, K. 2021. On the limited applicability of liquid democracy. In *Games, Agents, and Incentives Workshop*.

Awan, K. A.; Din, I. U.; Almogren, A.; Guizani, M.; and Khan, S. 2020. StabTrust—A stable and centralized trust-based clustering mechanism for IoT enabled vehicular ad-hoc networks. *IEEE Access*, 8: 21159–21177.

Behrens, J.; Kistner, A.; Nitsche, A.; and Swierczek, B. 2021. The temporal dimension in the analysis of liquid democracy delegation graphs. *Interaktive Demokratie*.

Bentert, M.; Boehmer, N.; Rymar, M.; and Tannenberg, H. 2022. Who won? Winner determination and robustness in liquid democracy. *arXiv preprint arXiv:2205.05482*.

Birmpas, G.; Lazos, P.; Markakis, E.; and Penna, P. 2024. Reward schemes and committee sizes in proof of stake governance. *arXiv preprint arXiv:2406.10525*.

Bloembergen, D.; Grossi, D.; and Lackner, M. 2019. On rational delegations in liquid democracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1796–1803.

Blum, C.; and Zuber, C. I. 2016. Liquid democracy: Potentials, problems, and perspectives. *Journal of Political Philosophy*, 24(2): 162–182.

Boldi, P.; Bonchi, F.; Castillo, C.; and Vigna, S. 2009. Voting in social networks. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 777–786.

Brill, M.; Delemazure, T.; George, A.-M.; Lackner, M.; and Schmidt-Kraepelin, U. 2022. Liquid democracy with ranked delegations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4884–4891.

Campbell, J.; Casella, A.; de Lara, L.; Mooers, V. R.; and Ravindran, D. 2022. Liquid democracy. Two experiments on delegation in voting. Technical report, National Bureau of Economic Research.

Colley, R.; Delemazure, T.; and Gilbert, H. 2023. Measuring a priori voting power in liquid democracy. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2607–2615.

Dasgupta, S.; Papadimitriou, C. H.; and Vazirani, U. 2006. *Algorithms*. McGraw-Hill, Inc.

Escoffier, B.; Gilbert, H.; and Pass-Lanneau, A. 2019. The convergence of iterative delegations in liquid democracy in a social network. In *Proceedings of the International Symposium on Algorithmic Game Theory*, 284–297.

Escoffier, B.; Gilbert, H.; and Pass-Lanneau, A. 2020. Iterative delegations in liquid democracy with restricted preferences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1926–1933.

Faliszewski, P.; Hemaspaandra, E.; and Hemaspaandra, L. A. 2011. Multimode control attacks on elections. *Journal of Artificial Intelligence Research*, 40: 305–351.

Faliszewski, P.; and Rothe, J. 2016. Control and Bribery in Voting. In *Handbook of Computational Social Choice*, 146–168. Cambridge University Press.

Gersbach, H.; Mamageishvili, A.; and Schneider, M. 2022. Staking pools on blockchains. *arXiv preprint arXiv:2203.05838*.

Gölz, P.; Kahng, A.; Mackenzie, S.; and Procaccia, A. D. 2021. The fluid mechanics of liquid democracy. *ACM Transactions on Economics and Computation*, 9(4): 1–39.

Green-Armytage, J. 2015. Direct voting and proxy voting. *Constitutional Political Economy*, 26: 190–220.

Grossi, D.; Hahn, U.; Mäs, M.; Nitsche, A.; Behrens, J.; Boehmer, N.; Brill, M.; Endriss, U.; Grandi, U.; et al. 2024. Enabling the digital democratic revival: A research program for digital democracy. *arXiv preprint arXiv:2401.16863*.

Hemaspaandra, E.; Hemaspaandra, L. A.; and Rothe, J. 2007. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, 171(5-6): 255–285.

Kahng, A.; Kotturi, Y.; Kulkarni, C.; Kurokawa, D.; and Procaccia, A. 2018. Ranking wily people who rank each other. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1087–1094.

Kahng, A.; Mackenzie, S.; and Procaccia, A. 2021. Liquid democracy: An algorithmic perspective. *Journal of Artificial Intelligence Research*, 70: 1223–1252.

Kling, C.; Kunegis, J.; Hartmann, H.; Strohmaier, M.; and Staab, S. 2015. Voting behaviour and power in online democracy: A study of LiquidFeedback in Germany's Pirate Party. In *Proceedings of the International AAAI Conference on Web and Social Media*, 208–217.

Markakis, E.; and Papasotiropoulos, G. 2021. An approval-based model for single-step liquid democracy. In *Proceedings of the International Symposium on Algorithmic Game Theory*, 360–375.

Miller III, J. C. 1969. A program for direct and proxy voting in the legislative process. *Public Choice*, 7(1): 107–113.

Mishra, M.; Gupta, G. S.; and Gui, X. 2019. Trust-based cluster head selection using the k-means algorithm for wireless sensor networks. In *Proceedings of the International Conference on Smart Systems and Inventive Technology*, 819–825.

Paulin, A. 2020. An overview of ten years of liquid democracy research. In *Proceedings of the International Conference on Digital Government Research*, 116–121.

Tullock, G. 1967. Proportional Representation. *Toward a Mathematics of Politics*, 144–157.

Zhang, Y.; and Grossi, D. 2021. Power in liquid democracy. In *Proceedings of the AAAI conference on Artificial Intelligence*, 5822–5830.