The Communication Burden of Single Transferable Vote, in Practice

Manel Ayadi, Nahla Ben Amor and Jérôme Lang

Abstract

Single Transferable Vote (STV) is of particular interest for voting, for cognitive reasons (it is easy to understand) and normative reasons (it is clone-proof). However, assuming that voters have to report full rankings sometimes makes it highly unpractical. We study single-winner STV from the point of view of communication. In the first part of the paper, we assume that voters give, in a single shot, their top k alternatives; we define a version of STV that works for such k-truncated votes, and we evaluate empirically (on randomly generated profiles, and on real data) the extent to which it approximates the standard STV rule. In the second part of the paper, we start from the protocol used by Conitzer and Sandholm (2005) for assessing the communication complexity of STV. We give an improvement of it, and then we study empirically the average communication complexity of these protocols, based on the one hand on randomly generated profiles, and on the other hand on real data. We also first give a polynomial-time computable characterization of possible winners at each step of this protocol. Our conclusion is that STV needs, in practice, much less information than in the worst case.

1 Introduction

There has been a significant attention these last fifteen years about incomplete information and communication issues on voting; the most complete reference to date on the topic is the handbook chapter [4]. There are (at least) two paths of work in this area: one focusing on computing partial or approximate outcomes from partial information about the votes, and the other one focusing on communication protocols. For the first path, one natural solution to the impracticality of submitting complete orders is to allow the voters to cast *truncated ballots*. The advantage of doing so is that not only it saves communication effort, but it is also often easier for voter to submit a partial ranking when the total amount of preference information is too large to communicate.

We focus on a specific, but particularly important, single-winner voting rule: *single transferable vote* (STV).[1] Out of all the common voting rules of the zoo, STV is among the few ones that is used in political elections, both for single-winner voting and multi-winner voting.[2] There are good reasons for that. First, it is relatively easy to understand. Second, it is hard to manipulate. Third, it enjoys a very important normative property: clone-proofness. On the other hand, it fails to satisfy a number of other important properties, such as monotonicity, participation, or Condorcet-consistency, but in many contexts, being sensitive to cloning balances the failure of these other properties. To give a simple example: at the 2002 French presidential, it has been argued that sensitivity to cloning was the cause of the socialist candidate Lionel Jospin not to go to the second round (which he would probably have won). Sensitivity to cloning is of primary importance not only for political elections, but also for low-stake collective decision making [19, 10].

On the other hand, when compared to other rules that are widely used in practice (such as plurality, k-approval for small k, approval, or plurality with runoff), STV suffers from a significant drawback: its direct implementation requires an important amount of information to be communicated from the voters in order the outcome to be determined, because its input consists of a collection of complete rankings over candidates. Our aim is to get a more accurate idea of the precise amount of

[1] For single-winner elections, STV is often called *instant runoff voting* (IRV). We keep however the terminology STV, which seems to more popular among the community, even for single-winner elections.

[2] See `https://en.wikipedia.org/wiki/Single_transferable_vote`.

information that we need from the voters to compute or approximate STV. We consider successively two contexts, and their associated questions.

For the first context, we assume that the information from the voters has to be communicated in a single shot. In this context, computing the exact winner may need, in the worst case, all the voters' rankings. We suggest instead to use *truncated ballots*: each voter reports only their top k candidates, for a fixed (voter-independent) k, an we use an approximation of STV which needs only these top-k ballots as input. The key question is then: *how often will this approximation of STV lead a mistake and output a different winner than we would have obtained with complete ballots under STV?*

Using truncated ballots as a way of reducing the amount of information in voting has been considered in a few earlier works, especially [2, 20, 18, 9, 17]. In the same spirit, we focus on STV and show that, *in practice* (on randomly generated profiles using classical distributions, and on real data), even for a small value of k, the approximation of STV outputs the correct winner quite frequently. We complete our study by examining how frequently the approximation of STV will be sensitive to cloning, and how often the winner will coincide with the winner of plurality with runoff.

For the second context, we consider *interactive elicitation*: a communication protocol is run between the central authority and the voters, until the outcome of the vote is eventually determined. The goal is to find a cheap protocol, where the cost of a protocol is the worst-cased number of bits transmitted. The (deterministic) communication complexity of a voting rule is the minimum, over all protocols that compute this rule, of the cost of the protocol.

This line of research has been initiated by Conitzer and Sandholm [6], who study the communication complexity of STV (among other common voting rules) and gave a (worst-case) almost optimal protocol. For other rules than STV, Kalech et al. [11] give a protocol that proceeds in rounds, considering top-k ballots with increasing k until the winner is determined. Lu and Boutilier [14, 13] and Dery et al. [7] also study elicitation protocols based on top-k ballots.

We start from Conitzer and Sandholm's protocol [6]. We give a practical improvement of it. These protocols work by asking, at each step, some selected voters to give their next preferred candidate among some specified set of candidates, when their currently preferred candidate has just been eliminated. We give a characterization of possible winners in that particular context (that is, candidates that win for at least one way of completing the current truncated ballots). The next question we are interested in is *what is the sufficient number of bits (on average) that must be communicated by the voters to the central authority so that the winner is determined?* We study empirically the average communication complexity of these protocols, based on the one hand on randomly generated profiles, and their practical communication complexity, based on real data sets from *PrefLib*.

The remainder of this paper is structured as follows: Section 2 gives basic background on voting. Section 3 defines the k-truncated approximation of STV and addresses its empirical evaluation. Section 4 focuses on communication protocols for STV, studies their communication complexity in average and in practice, and characterizes possible winners along the execution of the protocols.

2 Preliminaries

2.1 Single Transferable Vote

An election is a triple $E = (N, A, P)$ where: $N = \{1, ..., n\}$ is the set of *voters*, $A = \{a, b, ...\}$ is the set of *candidates*, with $|A| = m$; and $P = (\succ_1, ..., \succ_n)$ is the *(preference) profile* of voters in N, where for each i, $\succ_i \in P$, the preference relation of voter i, is a linear order over A. P is said to be complete if and only if $\succ_i$ is complete for each $0 \leq i \leq n$. (We will refer to this order either as a *preference order* or a *vote*; we use these terms essentially interchangeably.)

An irresolute voting rule is a function which, for each election, outputs a non-empty subset of A, whose elements are called the (co-)winners. A resolute voting rule is a function $f : E \mapsto A$, which for each election outputs a single winner.

Given a prespecified linear order $\triangleright$ over the candidates, called *tie-breaking priority*, the $STV^{\triangleright}$ rule proceeds in (up to $m-1$) rounds. (For brevity notation we will simply write STV, leaving $\triangleright$ implicit.) In each round, the candidate with the smallest number of voters ranking them first is eliminated (using the tie-breaking priority if necessary),[3] and the votes who supported it now support their preferred candidate among those that remain.

STV is known to be *clone-proof* [21]: whenever a candidate x is cloned into several candidates (ranked contiguously in every vote), then either the winner was x, in which case the winner in the profile where x has been cloned is one of the clones of x; or the winner was $y \neq x$, and in that case the winner after x has been cloned is still y. Most voting rules with ordinal input fail to be clone-proof.[4]

Freeman *et al.* [10] give an axiomatisation of STV, and show that STV is the only rule in a large family of iterative elimination rules that satisfies clone-proofness. STV is NP-hard to manipulate, even for one voter [1], although this worst-case difficulty does not really carry on to real-world instances [22].

2.2 Incomplete preferences and communication protocols in voting

When the votes are partly known, the outcome can generally not be determined. In this case, a candidate c is a *possible winner* for partial profile P (considered to be a collection of partial votes, that is, partial orders over candidates) and voting rule f if c wins for at least one complete extension of the partial votes in P. A particularly intuitive way of expressing partial preferences is to make use of *truncated ballots*. Given $k \leq m$, a *top-k ballot* is a linear order of k among the m candidates in A. A *top-k profile* is a collection of n top-k ballots.

A voting rule does not specify how the votes are elicited from the voters by the central authority (these rules are just functions mapping the preferences of all the voters to a winner). Different ways of determining the winner of an election by eliciting preferences from voters are based on specific *protocols*. A protocol is similar to an algorithm, with instructions replaced by communication actions; such actions specify bits that the voter should communicate, depending on her knowledge [12]. Formally, a *protocol for a voting rule f* is a protocol that computes $f(\succ_1, ..., \succ_n)$, given that $\succ_i$ is the private information of voter i. The *(deterministic) communication complexity of a voting rule f* is the minimum cost of a protocol for f. At each step of a protocol, the voters have reported a partial information about their vote (consisting of the sequence of bits they have sent until then). Therefore, the notion of possible winner makes sense at any step of a voting protocol.

3 Approximating STV with Truncated Ballots

3.1 STV_k

For each $k \leq m$, STV_k is defined similarly as STV, but with top-k ballots as input. In each round, the candidate ranked first by the smallest number of voters is eliminated (using tie-breaking if needed). The difference with STV is that when all k candidates in a vote have been eliminated, the vote is ignored in later rounds (such a vote will be said to be *exhausted*). We repeat this process until there exists a candidate ranked first by the majority of non-exhausted truncated votes. STV_1 coincides with plurality, and STV_{m-1} (and STV_m) with STV.

In this Section we consider only *one-shot protocols* where all the information of a voter is transmitted once.

[3] This way of breaking ties in STV is called *immediate tie-breaking*. One may choose instead to consider all the possible ways of breaking ties, and then use the tie-breaking in the very end to choose between the winners obtained this way: this is called "parallel universe" STV [5]. We will not consider it here.

[4] Apart of STV, other noticeable exceptions are *Ranked Pairs* and *Schulze*. Among rules where the input does not consist of rankings over candidates, we find other clone-proof rules, especially *approval voting* and *range voting*.

Example 1. *Let $A = \{a, b, c, d, e\}$ and the following 21 top-2 ballots: 6 votes $a \succ e$, 5 votes $d \succ e$, 4 votes $c \succ e$ and 6 votes $b \succ c$, with tie-breaking priority $a \rhd b \rhd c \rhd d \rhd e$. Under STV_2, e is eliminated first, then c. The votes $c \succ e$ are now exhausted, and will be ignored. d is eliminated next; a and b are then tied, and a is the STV_2 winner thanks to tie-breaking.*

Although STV_k can be seen as a voting rule on its own, we choose to see it as an approximation of STV and we evaluate its accuracy in this context. For this we start by defining voting distributions on which our study will be based. We focus on the *Mallows ϕ* model [15] because of its flexibility and ability to represent a wide class of preferences (including impartial culture when $\phi = 1$). We will also discuss experiments using real data sets.

When a voting rule is defined via the maximization (or minimization) of a score, it makes sense to measure the quality of its approximations using score ratios. However, STV is not based on score maximization (see [5] for a discussion), and the only reasonable way we see of measuring the quality of an approximation is to measure the frequency with which the approximation outputs the true winner. Our main practical objective is to obtain, depending on the parameters of the setting, the minimal value of k such that the probability of obtaining the true winner from STV_k is high enough.

As a side result, mainly out of curiosity, we will study the probability that STV_k outputs the same winner as plurality with runoff, depending on k. Finally, we study empirically the resistance of STV_k to cloning, given that we already know that for $k = 1$, STV_1 coincides with plurality, which is highly sensitive to cloning, and for $k = m - 1$, with STV, which is clone-proof. We will see that, while in theory STV_k fails to satisfy clone-proofness for each $k \leq m - 2$, in practice, it resists quite well to cloning even with small values of k.

Next, we consider real data sets (n, m) with small and large number of voters from *Preflib* [16]: *Sushi* (5000,10), *Dublin* (3662,12), *Electoral Reform Society(ERS)* (43,10), *Glasgow City Council* (548,9) and *Debian 2005* (327,7).

3.2 Evaluation of the Accuracy of STV_k

In order to measure experimentally the probability that the output of the STV_k coincides with the true STV winner, we repeatedly do the following: (1) generate a complete profile P (with n voters and m candidates); (2) for $k = 1, \ldots, m - 2$, we compare $STV_k(P)$ to $STV(P)$. Step (1) varies according to whether we are in the random generation setting or the real world data setting. For the former, we draw a profile according to a given distribution. For the latter, we draw a profile by selecting n votes at random in the database. These two steps are then iterated a sufficient number of times so as to obtain meaningful results.[5] Iterating this process a number of times allows us to evaluate the quality of the *top-k* approximation for different values of k.

3.2.1 Mallows ϕ

In this set of experiments, we use realistic parametrized families of probability distributions over rankings, namely, the *Mallows ϕ-model* [15], parametrized by a modal or reference ranking σ and a dispersion parameter $\phi \in [0, 1]$, and for any ranking r we define the probability of selecting r given σ and ϕ as follows: $P(r; \sigma, \phi) = \frac{1}{Z}\phi^{d(r,\sigma)}$, where d is the Kendall tau distance and $Z = \sum_{r'} \phi^{d(r,\sigma)} = 1 \cdot (1 + \phi) \cdot \left(1 + \phi + \phi^2\right) \cdot \ldots \cdot \left(1 + \ldots + \phi^{m-1}\right)$ is a normalization constant. With small values of ϕ, the mass is concentrated around σ, while $\phi = 1$ gives the uniform distribution *impartial culture (IC)*, where all profiles are equiprobable.

[5] Note that we cannot simply stop the process if we reach a k such that $STV_k(P) = STV(P)$, because there is no guarantee that $STV_{k'}(P)$ will still be equal to $STV(P)$ for $k' > k$. For instance, consider the profile composed of 5 votes $b \succ a \succ c \succ d$, 4 votes $a \succ b \succ c \succ d$, 3 votes $c \succ d \succ b \succ a$ and one vote $d \succ a \succ b \succ c$: the STV winner is b. For STV_1, it is b too. However, for STV_2, it is a.

For each experiment, we draw 1000 random profiles. In the first set of experiments, we present simulation results with $m = 7$ candidates and varying the number of voters n and the dispersion parameter ϕ. We simulate the elicitation of *top-k* ($k \in \{1 \dots 6\}$) preferences for $n = \{100, ..., 500\}$ with different values of $\phi \in \{0.7, 0.8, 0.9, 1\}$. Figure 1 shows results reflecting success probabilities for which eliciting top-k truncated ballots could predict the true winner.

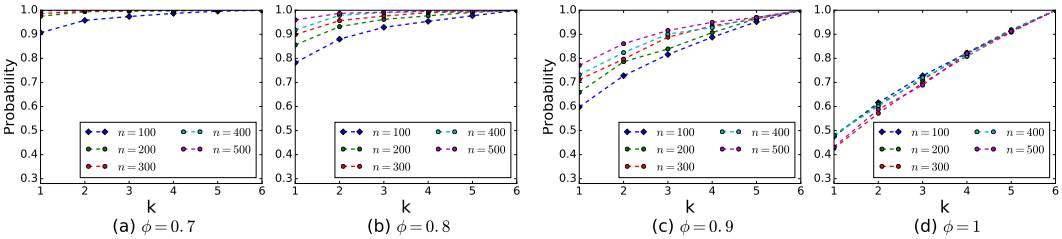


Figure 1: Success probabilities of *top-k* voting for STV_k: $m = 7$ varying n, k and ϕ.

From the results, we see that when $\phi \leq 0.7$ and $n \geq 500$, the true winner is always predicted correctly with all values of k. For $n \geq 200$ and $\phi \leq 0.7$, top-2 truncated ballots are always sufficient to determine the correct STV winner. As we increase the value of dispersion parameter $\phi \in \{0.8, 0.9\}$, good results are depicted with a great number of voters i.e. For $\phi = 0.8$ (resp. $\phi = 0.9$) and $n \geq 400$ (resp. $n \geq 500$), the accuracy reaches 99% (resp. 95%) with top-3 (resp. top-4) truncated ballots. When $\phi = 1$, the success rate is 82% (resp. 91%) when considering top-4 (resp. top-5) truncated ballots of 500 voters. In all cases, *top-2* truncated ballots seems to be always sufficient to predict the correct STV winner with 100% accuracy with small values of ϕ and high number of voters.

3.2.2 Real Data

With all real data sets, STV_k when $k \in \{1, \dots, m-1\}$ converges quickly to the correct prediction with 100% accuracy with all values of k for increasing n. Now we are interested in predicting the result for small and large elections. We consider Dublin data with samples of n^* voters among n ($n^* < n$), starting by $n^* = 10$ voters and incrementing each time the number of voters by 10. In each experiment, 1000 random profiles are constructed with n^* voters; then we consider the top-k ballots of these latter, where $k \in \{1, 2, 3\}$, and we compute the probability of selecting the correct winner (the winner of the complete profile of the n^* sampled votes). Figure 5 shows results for Dublin data with small ($n^* \in \{10, ..., 100\}$) and large elections ($n^* \in \{110, ..., 2000\}$).

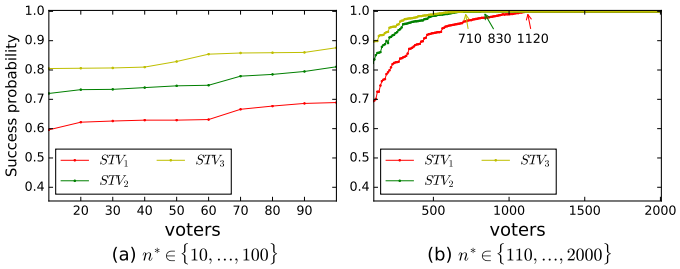


Figure 2: Success probabilities of STV_k with Dublin data: varying $k \in \{1, 2, 3\}$ and n^* ($n^* < n$).

Our results suggest that predicting the correct winner with a small number of voters fails significantly often when k is too small ($k \leq \frac{1}{4}m$). For instance, when $k = 3$ and $n^* = 100$, the correct winner is predicted only with frequency 87%. We also see that performance increases with the number of voters. Indeed, eliciting one candidate for each voter ($k = 1$) is sufficient to predict the correct

winner when $n^* \geq 1120$. Obviously, increasing the value of k leads to a decrease in the number of voters needed for correct winner selection with 100% accuracy: for instance, when $k = \frac{1}{6}m$ (resp. $k = \frac{1}{4}m$) over 12 candidates, $n^* \geq 830$ (resp. $n^* \geq 710$) are needed to always output the correct result. We simulated the same experiments on Sushi data, we find that a very small number of voters ($n^* \geq 160$) reporting their top-1 ballots are sufficient to determine the winner correctly.

3.3 Plurality with Runoff and STV_k

Plurality with runoff is a widely used rule that proceeds in two rounds: the two candidates ranked first by the largest number of voters go to the second round, and majority is used to determine the winner. STV and plurality with runoff belong to the same family of elimination-based rules. Also, when $m = 3$, STV coincides with plurality with runoff. On the other hand, plurality with runoff also shares a lot with plurality. Now, our family of STV$_k$ rule has plurality at one extremity and STV at the other extremity. It is therefore an interesting question (at least out of curiosity) to know where in this spectrum we have an output that is likely to coincide with the output of plurality with runoff.

We answer this question empirically. For each experiment, we generate 1000 random profiles according to a Mallows distribution, for $m = 7$ candidates, and we vary the number of voters $n \in \{100, ..., 500\}$ and the dispersion parameter $\phi \in \{0.9, 1\}$. We simulate the elicitation of *top-k* ($k \in \{1 \dots 6\}$) preferences where for each value of k, we compare the winner of STV_k with the winner of plurality-with-runoff. Figure 3 shows the probability that the STV_k's winner coincides with the winner of plurality with runoff.

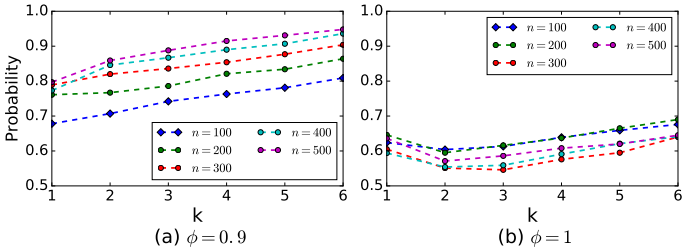


Figure 3: Probability that STV_k coincides with plurality with runoff: $m = 7$, vary n, k and ϕ.

With small values of ϕ (i.e., $\phi \leq 0.8$) and $k = 2$, STV_k and plurality with runoff always coincide, which is explained by the fact that the distributions are concentrated then the winner is the same for most common rules. Unsurprisingly, with greater values of dispersion parameter $\phi \geq 0.9$, the probability decreases, since all profiles are equiprobable, and good results are obtained with large number of voters. Perhaps more surprisingly, for impartial culture, the closeness between STV_k and plurality with runoff starts by decreasing from $k = 1$ to $k = 2$ or $k = 3$ (depending on n) and beyond that, increases with k. We do not observe this phenomenon with smaller values of ϕ.

3.4 STV_k and Resistance to Cloning

While STV is clone-proof, unfortunately this does not carry on to STV_k for $k \leq m - 2$ (and for $k = 1$, STV_1 is plurality, which is extremely sensitive to cloning). We may wonder whether for sufficiently large values of k, the occurrences of profiles in which cloning makes a difference for STV_k is small enough.

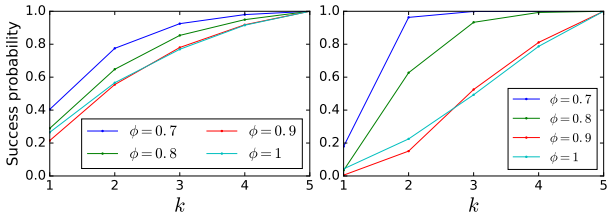
In order to evaluate the resistance of STV_k to cloning, we propose an empirical approach where we clone one candidate and we measure experimentally the probability that cloning significantly changes the outcome. For doing so we repeatedly generate a complete profile P (with n voters and m candidates), and then for each $k \in \{1, \dots m\}$ (a) we construct a profile P' obtained from P by cloning a candidate (note that there are $m + 1$ candidates in P'), and (b) we compare

$STV_k(P)$ to $STV_k(P')$. These steps are then iterated a sufficient number of times so as to obtain meaningful results.

Example 2. *Let P contain 4 votes $a \succ c \succ b$, 3 votes $b \succ a \succ c$, and 2 votes $c \succ b \succ a$. The STV_2 (= STV) winner is b. Let us clone c in $\{c, c'\}$. A possible profile P' obtained is $\{a \succ c \succ c' \succ b, c' \succ c \succ b \succ a, b \succ a \succ c \succ c'\}$. For $k = 2$, the k-truncated profile is $\{a \succ c, c' \succ c, b \succ a\}$ and the STV_2 winner is a.*

3.4.1 Mallows ϕ

For each experiment we draw 1000 profiles. We present simulation results for small and large elections when $m = 5$ as we vary ϕ. We simulate the elicitation of top-k preferences where $k \in \{1, \ldots, 5\}$ for $n \in \{30, 500\}$ with different values of $\phi \in \{0.7, 0.8, 0.9, 1\}$. We say that cloning sensitivity does not occur if the winner after cloning is the same as before, or a clone of the winner before cloning.

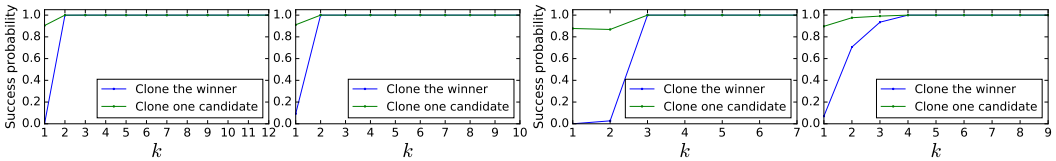


(a) $n = 30, m = 5$ (b) $n = 500, m = 5$

Figure 4: Resistance to cloning, winner cloned.

We give the probability that clone sensitivity does not occur under STV_k, when we clone the STV_k winner (Figure 4). Unsurprisingly, resistance to cloning increases rapidly with k and decreases with ϕ. Also, it significantly increases with the number of voters.

3.4.2 Real Data



(a) *Dublin* (b) *ERS* (c) *Debian* (d) *Glasgow city*

Figure 5: Resistance to cloning: winner and one candidate at random cloned.

We test the resistance of STV_k to cloning using real data sets. In a first series of experiments, we clone one random candidate; in a second one, we clone the STV_k winner (Figure 5). Again, sensitivity to cloning increases rapidly with k, even more rapidly than when randomly generated profiles.

4 Communication Protocols for STV

We now take a different path: we now allow for more sophisticated, *interactive* protocols where voters may report their preferences incrementally, when the central authority asks them to do so; on the other hand, we are not any longer interested in computing an approximation of STV, but in computing the real STV winner.

4.1 Conitzer and Sandholm's Protocol

With the aim of assessing the communication complexity of STV, Conitzer and Sandholm [6] give the following protocol for STV, illustrated in Protocol 1 which we call P_1.

Protocol 1: P_1

foreach *voter* i, $i \in N$ **do**
  $P = (a_1, ..., a_n)$ where a_i contains only the top candidate of each voter i
repeat
  $d \leftarrow$ candidate ranked first by the fewest voters
  Remove d from the set of available candidates ($A \leftarrow A \setminus \{d\}$)
  foreach *voter* i, $i \in N$ **do**
    if *the preference* a_i *of voter* i *starts with* d **then**
      $a_i \leftarrow$ Next preferred candidate of voter i
  Votes for d transferred to the next best remaining candidate
until *There exists a candidate* d *ranked first by a majority* ***or*** $|A| = 1$

In [6] it is shown that the communication complexity of P_1 is in $O(n(\log m)^2)$. The key argument is that at a step where there remain k candidates, the number of voters who rank the eliminated candidate first is at most $\frac{n}{k}$, therefore, the number of such messages received by voters is at most $\frac{n}{m} + \frac{n}{m-1} + \ldots + n$, which is in the order of $n \log m$. We can actually say something more precise : the worst case occurs when, in each step, all candidates are tied for elimination; in this case, when k candidates remain, exactly $\frac{m}{k}$ voters will send $\log k$ bits, and this goes on until $k = 3$. This gives an exact worst case cost of $n\left(\log m + \sum_{k=1}^{m-1} \frac{\log k}{k+1}\right)$. However the actual cost can be significantly smaller for small values of m: for instance, for $m = 7$, the real worst-case cost is approximately $4.69n$ whereas $(\log 7)^2 n \approx 7.88n$.[6]

Example 3. *Let* $A = \{a, b, c, d\}$ *and the following eight votes* $P = \{3 : a \succ c \succ d \succ b, 3 : c \succ a \succ b \succ d, 1 : d \succ b \succ a \succ c, 1 : b \succ d \succ c \succ a\}$ *with tie-breaking priority* $a \rhd b \rhd c \rhd d$. *Following* P_1, *given the top candidate of each vote,* d *is eliminated in the first round. We ask the voter who supports* d *for her next preference among* $A = \{a, b, c\}$. *Now,* $P = \{3 : a, 3 : c, 1 : b, 1 : b\}$ *and* b *is eliminated next. Then, we ask the voters supporting* b *for their next preference i.e.* $P = \{3 : a, 3 : c, 1 : a, 1 : c\}$. *Finally,* a *wins thanks to the tie-breaking priority.*

4.2 Possible Winners along Conitzer and Sandholm's Protocol

At each step in the execution of the protocol, the central authority has partial knowledge of the votes. Therefore, it makes sense to identify those candidates that can still win (the possible winners) and those that cannot (the necessary losers). This is especially useful when interaction with the voters takes time: assume that the vote is about a meeting date; the execution of the protocol can take several days (due to some voters reacting slowly to their emails). If at some point in the execution of the protocol, we know that for sure the meeting will not be on November 22 nor November 24, this is useful information for voters, who can plan something else on these two days, and for the central authority, which does not have to pre-book a room for these days.

We start by noticing that at each step of the protocol, the reduced profile obtained after removal of the eliminated candidates is a *tops-only profile*: the central authority knows only the top candidate of each voter. Therefore, without loss of generality for computing possible winners along the protocol, it is enough to give a characterization of possible winners for STV and for tops-only profiles.[7]

[6] Note that we don't know whether this protocol is asymptotically optimal: the best known lower bound on the communication complexity of STV is $\Omega(n \log m)$ [6].

[7] Note that the characterization of *necessary* winners (winning for any possible completion of the current partial profile) is trivial: x is the (only) necessary winner if it is top-ranked by a majority of voters, or if it is top-ranked by exactly half of the votes and has priority over other remaining candidates.

Proposition 1. *Let P be a tops-only profile over a set of candidates A. Let $S(x, P)$ be the plurality score of x in P, that is, the number of votes in P ranking x on top. Consider the following algorithm:*

1. *reorder the candidates in A in ascending order of $S(., P)$, breaking ties when necessary (lower-priority candidates being listed before those with higher priority). Let $x_1, \ldots, x_m$ be the obtained reordering of the candidates.*
2. *For each $i \leq m$ do*
   (a) $S_{max}(x_i, P) \leftarrow \sum\{S(x_j, P), j \leq i\}$;
   (b) $k \leftarrow i + 1$
   (c) *While* $(S_{max}(x_i) > S(x_k)$ *or* $(S_{max}(x_i) = S(x_k)$ *and* $x_i \rhd x_k))$ *and* $k \leq m$ *do*
      i. $S_{max}(x_i, P) \leftarrow S_{max}(x_i, P) + S(x_k, P)$
      ii. $k \leftarrow k + 1$

Then x is an $STV_{\rhd}$ necessary loser for P if and only if there is some candidate $y \in A$ such that one of these two conditions is satisfied:

1. $S(y, P) > S_{max}(x, P)$, *and* $S(y, P) > S(x, P)$.
2. $S(y, P) \geq S_{max}(x, P)$, $S(y, P) > S(x, P)$, *and* $y \rhd x$.

Proof. The key element is that $S_{max}(x_i, P)$ is the maximum plurality score that x_i can obtain before being eliminated: as long as the condition of the loop is satisfied, in the best case where all votes for the candidates eliminated until x_{k-1} are transferred to x_i, then the next eliminated candidate is x_k, and x_i continues to the next round. Note that $S_{max}(x_m) = n$, since it is initialized to this value and the loop is not executed.

Assume (1) holds. Then all candidates z such that $S(z, P) < S(y, P)$ will be eliminated before z, because, as long as z has not been eliminated, the maximum value of $S(z, P_U)$ they can reach is $S_{max}(y, P)$. Therefore, x will be eliminated before y and is a necessary loser.

Assume (2) holds. The maximum value that $S(x, P_U)$ can reach is $S_{max}(y, P)$. If $S_{max}(y, P) > S(x, P)$, we are in case (1). Assume now $S_{max}(y, P) = S(x, P)$. If $S(x, P_U)$ never reaches $S_{max}(y, P)$, then x will be eliminated before y. If $S(x, P_U)$ reaches $S_{max}(y, P)$, then x will also be eliminated before y, because $y \rhd x$. Therefore, x is a necessary loser.

Finally, if neither (1) nor (2) holds, then letting $y = x_m$, we get $S_{max}(x, P) = n$, which means that in the best case (for x) where all votes for all eliminated candidates go to x, x will be the winner. □

Example 4. *Let P be such that $S(x_1, P) = 1$, $S(x_2, P) = 2$, $S(x_3, P) = 4$, $S(x_4, P) = 6$ and $S(x_5, P) = 12$. Then, the maximal scores of different candidates are as follows: $S_{max}(x_1, P) = 1$, $S_{max}(x_2, P) = 3$, $S_{max}(x_3, P) = S_{max}(x_4, P) = S_{max}(x_5, P) = 25$. Let us give a few comments: in the best case for x_2, the unique vote for x_1 is transferred to it; at the second round, it has 3 votes, but is eliminated right after that because $S(x_3, P) = 4 > 3$. In the best case for x_3, at the end of the second round, all votes for x_1 and x_2 have been transferred to it and it gets 7 votes. Since $7 > S(x_4, P) = 6$, in this case x_4 is eliminated next and if all its votes are transferred to x_3, then x_3 gets 13 votes and wins against x_5. In conclusion: x_1 and x_2 are necessary losers and x_3, x_4, x_5 are possible winners.*

As a corollary, possible winners for STV for tops-only ballots (and therefore, possible winners for STV along the Conitzer-Sandholm protocol) are polynomial-time computable, while the possible winner problem for STV is NP-complete in the general case, since it is NP-complete for the particular case of constructive manipulation [1]. What we do not know, however, is the complexity of the possible winner problem for STV and top-k ballots for a fixed (voter-independent) k.

4.3 Immediate Necessary Losers, and an Improved Protocol

Knowing who are the possible winners (and the necessary losers) at each step of the protocol is a useful information, *but* it is not a good idea to eliminate a candidate as soon as it becomes a necessary loser, because it can change the final outcome, as shown on the following example.

Example 5. *Let us consider an election with 10 voters and 5 candidates with the following profile* $P = \{5 : c \succ a \succ d \succ e \succ b, 1 : a \succ e \succ b \succ d \succ c, 2 : e \succ b \succ d \succ a \succ c, 2 : b \succ a \succ e \succ c \succ d\}$. *The winner for P is c.* $S_{max}(d, P) = 0$, $S_{max}(a, P) = 1$, $S_{max}(e, P) = 5$, $S_{max}(b, P) = 10$ *and* $S_{max}(c, P) = 10$, *therefore, the necessary losers given tops-only ballots are a, d and e. If we eliminate those three candidates, the winner is b.*

The reason is that although we know that a necessary loser x will be eliminated at some point before the end, we do now know exactly when, and depending on when it will be eliminated, the final winner can change. However, if we know exactly when it will be eliminated, then we can safely eliminate it during the execution of the protocol.

We now define a stronger notion of a necessary loser, which only applies to rules that proceed by sequential elimination: $x \in A$ is an *immediate loser* if we know that x will be the next candidate eliminated after the currently eliminated one. In Example 3, at the time where a is being eliminated, the central authority already knows that b will be the next eliminated candidate: b is an immediate loser, and it is useless to keep b in the candidate list when asking the voter who supported d who she ranks next. On the other hand, in Example 5, while a is a immediate loser when d is eliminated, e is not an immediate loser when a is eliminated: it can be the case that the voter supporting a transfers her vote to e, and b will then be eliminated before e.

Formally, let d be the candidate which is about to be eliminated, and U the set of remaining candidates (including d); candidate x is an *immediate loser* if for every $y \neq x, d$, either (1) $S(y, P_U) > S(x, P_U) + S(d, P_U)$, or (2) $S(y, P_U) = S(x, P_U) + S(d, P_U)$ and $y \rhd x$.

While eliminating a necessary loser in the course of the protocol may change the final outcome, *this can never be the case with an immediate loser*. This is the key property used in the next protocol, which is an improvement over protocol P_1: in Protocol 2 (which we call P_2), each voter first submits her most preferred candidate over the set of all available ones to the central authority (C) (lines 2-3); let d be the candidate with the fewest number of votes (with use of tie-breaking if necessary); if there is an immediate loser at this point, it is eliminated as well, together with d, from the set of available candidates. After d is eliminated, there may be *another* immediate loser; the process is repeated until there is no immediate loser (lines 9-12). Let IL be the set of the immediate losers in P_U.

After removing all immediate losers in P_U, we select a voter $i \in N$ such that her top-1 candidate a_i starts with d or $a_i \in IL$. We ask the selected voter to report her next preferred candidate among the available ones in U (lines 15-16). Indeed, when considering P_2, not all voters (with current best candidate among candidates in IL) are asked to submit their next preferred candidate after an immediate loser's elimination, at once. However, we select one voter at a time, we ask her to send her next preferred candidate and we re-compute the immediate losers given the new preference. We repeat this process (lines 8-15) until we obtain a top-only profile P with candidates among U for each voter. The advantage of doing so is that asking the voters one by one, can save communication costs since the new voter's preference may help to detect another immediate losers, thus reduce the set of available candidates. Finally, the process in lines (4-16) is repeated until there exists a candidate ranked first by more than 50% of the votes or only one candidate remains in the set of available candidates U.

Example 6. *We take the same profile as in Example 3. The initial tops-only profile gives 3 votes to a, 3 c, one to b and one to d. Using the tie-breaking priority, the first eliminated candidate is d; now, b is an immediate loser. Therefore, before querying anyone, we remove d* and *b. Then we select* one *voter among the two who supported b or d, assume we take the voter supporting d. She now supports a, and* there is no need to go further*: c is now an immediate loser and a is the winner.*

Protocol 2: P_2

Initialize: A, $IL = \emptyset$
foreach *voter* i, $i \in N$ **do**
  $P = (a_1, ..., a_n)$ where a_i contains only the top candidate of each voter i
repeat
  $d \leftarrow$ candidate ranked first by the fewest voters in P
  Remove d from the set of available candidates, $U \leftarrow A \setminus \{d\}$
  Remove d from all ballots in P (we obtain P_U)
  repeat
    repeat
      Remove the immediate loser x (if it exists) from $U \leftarrow U \setminus IL$ where, $IL = IL \cup \{x\}$
      Remove x from all ballots in P_U
    until *There is no immediate losers*
    Select a voter $i \in N$ where $a_i \in IL$ or $a_i = d$
    $a_i \leftarrow$ Next preferred candidate of voter i among candidates in U
  until *We have tops-only profile with candidates among* U *for each voter*
until *There exists a candidate ranked first by more than 50%* ***or*** $|U| = 1$

4.4 Evaluation of the Communication Protocols

This section presents the evaluation of the average communication complexity of P_1 and P_2. We discuss experiments using data generated from the Mallows ϕ model and real data. Our main practical objective is to determine the average communication complexity reported from voters in order to overturn the winner. We refer to P_{Worst} as the theoretical communication complexity.

4.4.1 Mallows ϕ

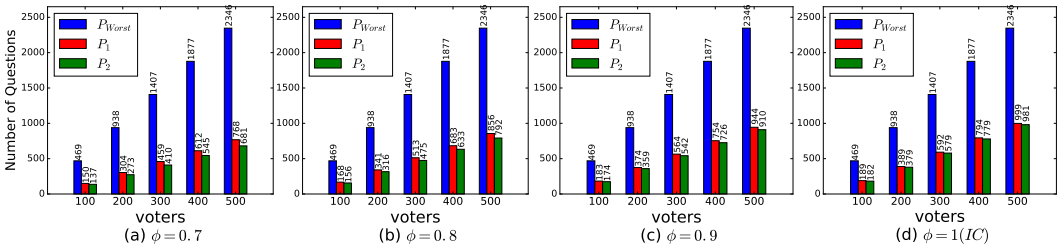


Figure 6: Average communication cost with P_1, P_2 and P_{Worst}

For each experiment, we draw 1000 random profiles. In the first set of experiments, we present simulation results with $m = 7$ candidates and let n and ϕ vary. We simulate the number of bits transferred between the central authority and the voters when $n = \{100, ..., 500\}$ with different values of $\phi = \{0.7, 0.8, 0.9, 1\}$. Figure 6 shows the average communication cost with P_1, P_2 and P_{Worst}. Results suggest that in practice, we can save a lot in communication costs for STV compared to the theoretical communication complexity. From the results, P_2 seems to be the best protocol to determine the STV winner with the lowest amount of information about preferences that is needed to accurately decide an election. Even with high dispersion parameter, using P_2 protocol, we can save almost 50% of bits communicated between the voters and the central authority e.g. when $n = 500$ and $\phi = 1$, only 981 bits are needed to determine the winner against 2346 bits in the worst case.

Now, we summarize the above results by measuring the normalized communication cost. Indeed, we compute the ratio of the number of bits elicited from voters using a specific protocol by the

number of bits elicited in the worst case. Figure 7 presents the obtained results as we vary ϕ.

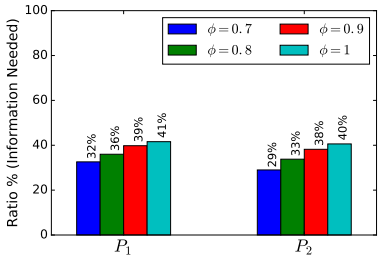


Figure 7: The percentage of communication needed with P_1 and P_2.

Depicted results suggest that with small values of ϕ, P_2 is efficient to reduce the communication cost e.g. for $\phi = 0.7$, only 29% (resp. 32%) of voters' preferences are needed to output the winner under P_2 (resp. P_1) protocol. As we increase ϕ ($\phi \geq 0.9$), more information is needed from voters and from the results we can detect that P_1 and P_2 become closer in communication cost e.g. when $\phi = 1$, 40% of preferences are elicited under P_2 against 41% when using P_1.

4.4.2 Real Data

Now, we propose to evaluate P_2 protocol using different real data sets. Figure 8 presents the average communication cost when considering data sets with small ($n < 1000$) and large ($n > 1000$) elections for P_1 and P_2. Then, Figure 9 summarizes the latter results by measuring the normalized communication cost. Consistent with the Mallows ϕ experiments, our results suggest that, in real vote situations, P_2 protocol presents the lowest communication cost for all real data sets. Indeed, when $n < 1000$, we save about 70% (resp. 77%) in communication when we use P_2 with Glasgow city and Debian data (resp. ERS). For $n > 1000$ and when we consider P_2, only 24% (resp. 26%) of preferences are needed to be communicated under Sushi (resp. Dublin) data.

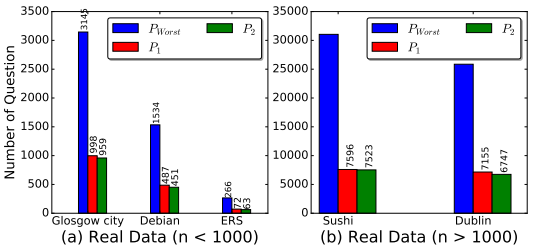


Figure 8: Average communication cost with P_1, P_2 and P_{Worst} using real data

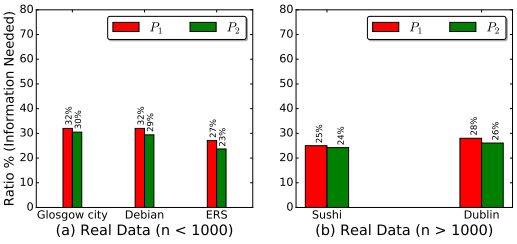


Figure 9: The percentage of communication needed with P_1 and P_2 using real data

5 Conclusion

In any context where applying STV would be a good idea if it was not unreasonable to ask voters to rank all candidates, one may think of using the STV$_k$ rules of Section 3, or the protocols of Section 4 (for those, one may think of implementing a smartphone application that sends a message to a voter each time her currently supported candidate has just been eliminated). An idea that we did not have time to develop is to consider STV in another incomplete information context, namely *vote streams*, where voters come one at a time in a streaming fashion (note that the recent work on vote streams [3, 8] do not consider STV); the key practical question is to decide when we have enough information to eliminate one more candidate, so that the next voters will have less information to communicate.

Acknowledgments. The authors are very grateful to the reviewers for providing useful and supportive feedback.

References

[1] John J. Bartholdi and James B. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8(4):341–354, Oct 1991.

[2] Dorothea Baumeister, Piotr Faliszewski, Jérôme Lang, and Jörg Rothe. Campaigns for lazy voters: truncated ballots. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*, pages 577–584, 2012.

[3] Arnab Bhattacharyya and Palash Dey. Fishing out winners from vote streams. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:135, 2015.

[4] Craig Boutilier and Jeffrey S. Rosenschein. Incomplete information and communication in voting. In *Handbook of Computational Social Choice*, pages 223–258. 2016.

[5] Vincent Conitzer, Matthew Rognlie, and Lirong Xia. Preference functions that score rankings and maximum likelihood estimation. In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pages 109–115, 2009.

[6] Vincent Conitzer and Tuomas Sandholm. Communication complexity of common voting rules. In *Proceedings of the 6th ACM conference on Electronic commerce*, pages 78–87. ACM, 2005.

[7] Lihi Naamani Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reaching a joint decision with minimal elicitation of voter preferences. volume 278, pages 466–487. Elsevier, 2014.

[8] Palash Dey, Nimrod Talmon, and Otniel van Handel. Proportional representation in vote streams. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 15–23, 2017.

[9] Yuval Filmus and Joel Oren. Efficient voting via the top-k elicitation scheme: a probabilistic approach. In *ACM Conference on Economics and Computation, EC '14, Stanford , CA, USA, June 8-12, 2014*, pages 295–312, 2014.

[10] Rupert Freeman, Markus Brill, and Vincent Conitzer. On the axiomatic characterization of runoff voting rules. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 675–681, 2014.

[11] Meir Kalech, Sarit Kraus, Gal A Kaminka, and Claudia V Goldman. Practical voting rules with partial information. volume 22, pages 151–182. Springer, 2011.

[12] Eyal Kushilevitz. Communication complexity. *Advances in Computers*, 44:331–360, 1997.

[13] Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *Proceedings of IJCAI International Joint Conference on Artificial Intelligence*, volume 22, pages 287–293, 2011.

[14] Tyler Lu and Craig Boutilier. Vote elicitation with probabilistic preference models: Empirical estimation and cost tradeoffs. In *Algorithmic Decision Theory*, pages 135–149. Springer, 2011.

[15] Colin L Mallows. Non-null ranking models. i. *Biometrika*, pages 114–130, 1957.

[16] Nicholas Mattei and Toby Walsh. Preflib: A library for preferences http://www. preflib. org. In *Algorithmic Decision Theory*, pages 259–270. Springer, 2013.

[17] Lihi Naamani-Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reducing preference elicitation in group decision making. *Expert Systems with Applications*, 61:246–261, 2016.

[18] Joel Oren, Yuval Filmus, and Craig Boutilier. Efficient vote elicitation under candidate uncertainty. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 309–316. AAAI Press, 2013.

[19] Markus Schulze. A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method. *Social Choice and Welfare*, 36(2):267–303, 2011.

[20] Piotr Skowron, Piotr Faliszewski, and Arkadii Slinko. Achieving fully proportional representation: Approximability results. *Artificial Intelligence*, 222:67–103, 2015.

[21] T. N. Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206, Sep 1987.

[22] Toby Walsh. An empirical study of the manipulability of single transferable voting. In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, pages 257–262, 2010.

Manel Ayadi
LARODEC, Institut Supérieur de Gestion (Université de Tunis)
Tunis, Tunisie
LAMSADE, Université Paris-Dauphine
Paris, France
Email: manel.ayadi@hotmail.com

Nahla Ben Amor
LARODEC, Institut Supérieur de Gestion (Université de Tunis)
Tunis, Tunisie
Email: nahla.benamor@gmx.fr

Jérôme Lang
LAMSADE, Université Paris-Dauphine
Paris, France
Email: lang@lamsade.dauphine.fr